Universidad de Santiago de Compostela

Departamento de Electrónica y Computación

TESIS DOCTORAL

SOFTWARE/HARDWARE FPGA-BASED SYSTEM FOR THE SOLUTION OF THE 3D HEAT EQUATION: APPLICATIONS ON THE NON-DESTRUCTIVE EVALUATION OF MINEFIELDS

Presentada por:
Fernando Rafael Pardo Seco

Dirigida por:
Diego Cabello Ferrer
Marco Balsi

Santiago de Compostela, Julio de 2008

Dr. **Diego Cabello Ferrer**,
Catedrático de Universidad del Área de Electrónica de la Universidad de Santiago de Compostela

Dr. **Marco Balsi**,
Ricercatore del Área de Ingegneria Elettronica de la Universitá La Sapienza de Roma

HACEN CONSTAR:

Que la memoria titulada **Software/hardware FPGA-based system for the solution of the 3D heat equation: applications on the non-destructive evaluation of minefields** fue realizada por D. **Fernando Rafael Pardo Seco** bajo nuestra dirección en el Departamento de Electrónica y Computación de la Universidad de Santiago de Compostela, y constituye la Tesis que presenta para optar al grado de Doctor en Ciencias Físicas.

Santiago de Compostela, Julio de 2008

Fdo: **Diego Cabello Ferrer**
Codirector de la tesis de doctorado

Fdo: **Marco Balsi**
Codirector de la tesis de doctorado

Fdo: **Javier Díaz Bruguera**
Director del Departamento de Electrónica y Computación

Fdo: **Fernando Rafael Pardo Seco**
Autor de la Tesis

Para Gemma y Fernando

Acknowledgements

I wish to express here my deep gratitude to all those who in one way or another have been involved in the development of this work.

I would like to thank my thesis advisor, Prof. Diego Cabello Ferrer for providing an excellent research environment and guiding me throughout all these years, with his insights, suggestions and invaluable advises.

I am also very grateful to Prof. Marco Balsi of the Dipartimento di Ingegneria Elettronica of Università "La Sapienza" and co-advisor of this thesis. His discusions about the hardware and the thermal model have been very helpful in the development of this work. I am also very thankful for guiding me in my first days at Rome and in the (difficult) search of a flat in that beautiful city.

I would like to acknowledge also to Prof. Ramón Ruiz for letting me join to his research group for 5 months at the Universidad Politécnica de Cartagena and treat me as one more of the group.

I would like to thanks very much to Ginés by his help in Cartagena and those grateful talks in the lab, for let me live in his new house, and above of all by his friendship.

Victor Brea has been a good support in my research thought all these years, his advises about electronic devices has been very good welcome.

Thanks to Paula by her help with the thermal model and by the help with the infrared images. Without your work this thesis couldn't have been written.

Digipi has given me a very good support every time I had a problem with Linux and other things, you know a lot. Thanks to Lebo, Rachel, Natalia and Pili, not everything is work, isn't it ?

Thanks to Pablo, Cristina, Lola by their friendship during my stay in Cartagena.

I would like to acknowledge Giancarlo for welcoming me in his house at Rome, for been my

guide in my first weeks in Italy, for those capuccinos in front of the Colosseo, for those weekends of football at the computer and for his friendship. I would also want to acknowledge Giuliano, Giulio, Mauro, Francesco and Marta for those good moments at Rome.

Thanks to Javi, Yago and the rest of the university basketball team for the good and not so good moments on the field

Thanks to Eduardo for those relaxing sport talks on Mondays. I would like to tanks to Roberto and David E. for those moments during our hard training for the races.

I would like to acknowledge my uncle Juan for his good advices during theses years.

Finally, I would like to acknowledge my wife Gemma and my child for their support during the elaboration of this work..

Julio de 2008

Who's the more foolish...the fool or the fool who follows him?

Obi-wan Kenobi in Star Wars a New Hope

Contents

Resumen **1**

Summary **13**

1 Non-destructive evaluation **17**

1.1 Introduction 17

1.2 NDE techniques 18

1.2.1 Visual inspection 20

1.2.2 X and gamma ray imaging 20

1.2.3 Liquid penetrant inspection 22

1.2.4 Ultrasonic testing 24

1.2.5 Electromagnetic inspection or eddy current inspection 26

1.2.6 Metal detector 28

1.2.7 Magnetic particle inspection 29

1.2.8 Ground penetrating radar 30

1.2.9 Infrared thermography 32

1.3 NDE techniques for buried landmines detection 34

1.3.1 IRT versus GPR 36

1.4 Infrared thermography for landmine detection 38
1.4.1 Physical phenomena 40
1.4.2 Thermal Model 48
1.4.3 Non-destructive soil inspection 50
1.5 Summary 60

2 FD-TD solvers **63**
2.1 Introduction 63
2.2 FD-TD method 63
2.3 Review of FD-TD hardware implementations 67
2.3.1 Schneider et al., 2002 67
2.3.2 Placidi et al., 2002 68
2.3.3 Durbano et al., 2003 68
2.3.4 Chen et al., 2004 70
2.3.5 Culley et al., 2005 72
2.3.6 He et al., 2005 73
2.3.7 Nagy et. al., 2006 75
2.3.8 Summary 75
2.4 FD-TD thermal model simulator 75
2.4.1 Algebraic Equations 79
2.4.2 Non-uniform media 83
2.4.3 Non-uniform grid 85
2.4.4 Non-uniform grid and media 89
2.5 Software-based FD-TD solvers 91

2.5.1 Reduction of size and duration of experiments 91
2.5.2 Efficient numerical methods 93
2.6 Summary 94

3 Heat equation hardware solver 95

3.1 Introduction 95
3.2 Hardware Description 96
3.2.1 Virtex-II FPGA 99
3.2.2 ZBT Memory Banks 106
3.3 FD-TD hardware solver 107
3.3.1 HOST 108
3.3.2 Hardware 111
3.4 FPGA implementation 114
3.4.1 Arithmetic analysis and wordlength 114
3.4.2 Hardware architecture 118
3.4.3 Processing element 121
3.4.4 Control Unit 128
3.4.5 Data flow 130
3.4.6 Characteristics of the implementation 154
3.5 Summary 154

4 Results 157

4.1 Introduction 157
4.2 Experimental setups 157
4.2.1 Indoor scenario 158

4.2.2 Outdoor scenario 159
4.3 Practical considerations 162
4.3.1 Resolution 162
4.3.2 Calibration 164
4.3.3 Initialization of the thermal model 165
4.3.4 Meteorological data 165
4.3.5 Moisture content 166
4.3.6 Thermal properties 166
4.4 Thermal model test 167
4.4.1 Full radiation test 167
4.4.2 Gradual heating test 174
4.4.3 Natural heating test 179
4.5 Classification example 185
4.6 Performance of the FGPA-based thermal model hardware solver 186
4.7 Summary 187

5 Conclusions and future work **199**

Index **205**

Glossary **207**

List of symbols **207**

Bibliography **209**

List of Figures

1.1 Different elements of a NDE system. 19

1.2 X-ray image showing holes caused by gas accumulation on a metal piece. 21

1.3 Different steps of the LPI technique 23

1.4 UT system to detect flaws. 25

1.5 Eddy current scheme to evaluate defects. 27

1.6 MPI applied to defect detection of magnetic materials. 30

1.7 Schematic of a GPR system. 32

1.8 Infrared image of the space shuttle in the landing step 33

1.9 A typical low metal antipersonnel mine (Type 72). 36

1.10 The electromagnetic spectrum. 41

1.11 Black body emissive power 43

1.12 Spectral distribution of solar radiation 45

1.13 Variation of sand reflectivity with the wavelength 46

1.14 Heat transfer processes 48

1.15 Mine detection approach flow 53

1.16 Sample of measured IR images of a minefield at sunrise. 54

1.17 Mine detection algorithm using the FPGA 60

2.1 Geometrical representation of the Yee cell. 66

2.2 One bit adder and subtractor used to perform the N-bit serial adder 68
2.3 Architecture of the system proposed by Placidi et. al. 69
2.4 Block diagram of the hardware implementation proposed by Durbano et. al. 69
2.5 Data path of the architecture proposed by Durbano et. al. 71
2.6 Hardware structure of Chen's FD-TD implementation. 72
2.7 Block diagram of Culley's implementation to update H_x, H_y and H_z. 73
2.8 System architecture of He et al. 74
2.9 Two dimensional grid showing spatial and temporal discretization. 77
2.10 Spatial discretization. 78
2.11 Heat flux contributions scheme from neighbor nodes for surface nodes. 79
2.12 Heat flux contributions scheme from neighbor nodes for internal nodes. 81
2.13 Heat conduction between adjoining and dissimilar materials (2D case). 84
2.14 Scheme of a discretization with a non-uniform grid in the z direction. 86
2.15 Scheme of non-uniform grid and media 90
2.16 Cylindrical symmetry of the buried landmine. 92

3.1 Standard FPGA structure. 97
3.2 Image of the RC2000 card. 98
3.3 RC2000 block diagram (Reprinted from RC2000 user manual). 100
3.4 Virtex-II architecture (Reprinted from Virtex-II datasheet). 101
3.5 Virtex-II IOB block (Reprinted from Virtex-II datasheet). 102
3.6 Virtex-II CLB element (Reprinted from Virtex-II datasheet). 102
3.7 Half part of a Virtex-II slice (Reprinted from Virtex-II datasheet). 104
3.8 Virtex-II slice configuration (Reprinted from Virtex-II datasheet). 105
3.9 Structure of a ZBT RAM from IDT (Reprinted from IDT ZBT RAM datasheet). . 108

3.10 General view of the system and connections between all components. 109

3.11 Handshaking protocol to share the use of the ZBT memory banks. 110

3.12 Variation of the trade-off with the number of bits. 119

3.13 Simulation with different bitlengths 120

3.14 Simulation using different bitlengths 121

3.15 Scheme used to load data from memory. 122

3.16 Processing element schematic: $PE1$... $PE6$. 125

3.17 Processing element schematic $PE0$. 126

3.18 Processing element schematic $PE7$. 127

3.19 Boundary block: circuit schematic that deals with the boundary conditions. 128

3.20 Chronogram of the control signals. 129

3.21 Considered neighbors to perform the updating process. 130

3.22 Temporal diagram of the read/write memory access 132

3.23 Scheme of the read/write operations from/to PEk (k=1...6). 133

3.24 Scheme of the read/write operations from/to $PE7$. 134

3.25 Scheme of the read/write operations from/to $PE0$. 135

3.26 Scheme of the read/write operations from/to $PE0$ processing surface nodes. 136

4.1 Experimental setup in indoor scenario. 158

4.2 Photo of the sand lane. 160

4.3 Ground truth of the sand lane - courtesy of TNO-FEL Laboratory. 161

4.4 Setup to obtain the IR images in the outdoor scenario. 163

4.5 Full radiation test: Values of the boundary conditions. 168

4.6 Full radiation test: double precision and integer arithmetic 169

4.7 Non-uniform grid using quadratic approach 171

4.8 Full radiation test: simulation using non-uniform grid 1 173
4.9 Full radiation test: simulation using non-uniform grid 2 174
4.10 Full radiation test: simulation using non-uniform grid 3 175
4.11 Values of the boundary conditions for the gradual heating test. 176
4.12 Gradual heating test: simulation using floating and integer arithmetic 178
4.13 Gradual heating test: simulation using non-uniform grid 1 180
4.14 Gradual heating test: simulation using non-uniform grid 2 181
4.15 Gradual heating test: simulation using non-uniform grid 3 182
4.16 Natural heating: IR image 183
4.17 Natural heating ex1: Initial temperature at 09:41. 183
4.18 Natural heating ex1: Temperature at 10:41 assuming mine absence 188
4.19 Natural heating ex1: Mask of detected targets 188
4.20 Natural heating ex1: Temperatures of the soil at 10:41. 189
4.21 Natural heating ex1: Temperature differences between MATLAB and FPGA . . . 189
4.22 Natural heating ex1: simulation at 10:41 using non-uniform grid 1 190
4.23 Natural heating ex1: simulation at 10:41 using non-uniform grid 2 191
4.24 Natural heating ex1: simulation at 10:41 using non-uniform grid 3 192
4.25 Natural heating ex2: Temperatures of the soil at 09:41 given by the IR camera. . . 193
4.26 Natural heating ex2: Temperatures of the soil at 10:41 193
4.27 Natural heating ex2: Temperature differences between MATLAB and FPGA . . . 194
4.28 Natural heating ex2: Simulation at 10:41 using non-uniform grid 1 195
4.29 Natural heating ex2: Simulation at 10:41 using non-uniform grid 2 196
4.30 Natural heating ex2: Simulation at 10:41 using non-uniform grid 3 197
4.31 Evolution of the error and the α value for $V82$ object. 198

List of Tables

2.1 Summary of the different FD-TD hardware solvers. 76

3.1 Different Virtex-II FPGA platforms and their main features. 106

3.2 SDK functions provided from Celoxica for the RC2000 card. 111

3.3 Handel-C functions to work with the RC2000 card. 112

3.4 Processing time for a clock frequency of 45 MHz. 113

3.5 Parameters that affect the trade-off of the system. 118

3.6 Empirically obtained bitlengths and scaling factor. 120

3.7 Data flow on $PE1$... $PE6$ (I). 140

3.8 Data flow on $PE1$... $PE6$ (II). 141

3.9 Data flow on $PE1$... $PE6$ (III). 142

3.10 Data flow on $PE7$ (I). 143

3.11 Data flow on $PE7$ (II). 144

3.12 Data flow on $PE7$ (III). 145

3.13 Data flow on $PE0$ (I). 146

3.14 Data flow on $PE0$ (II). 147

3.15 Data flow on $PE0$ (III). 148

3.16 Data flow on $PE0$ processing surface nodes (I). 149

3.17 Data flow on $PE0$ processing surface nodes (II). 150

3.18 Data flow on $PE0$ processing surface nodes (III). 151

3.19 Adders/Multipliers data flow on $PE0$ (I). 152

3.20 Adders/Multipliers data flow on $PE0$ (II). 153

3.21 FPGA utilization resources. 154

4.1 Indoor scenario: Physical characteristics of mines and soil 159

4.2 Outdoor scenario: Physical characteristics of the sandy soil. 159

4.3 Description of the AP and AT mines present in the sand lane. 162

4.4 Outdoor scenario: Description of non-mine objects present in the sand lane. 162

4.5 Thermal properties of the silicone rubber RTV3110. 163

4.6 QWIP camera specifications. 164

4.7 Full radiation test: differences between double and integer simulations 170

4.8 Times for different FPGA events. 170

4.9 Comparison of the simulations using uniform and non-uniform 172

4.10 Full radiation test: Results obtained using non-uniform grids. 176

4.11 Full radiation test: Results obtained using non-uniform grids II 177

4.12 Gradual heating test: Results obtained using non-uniform grids. 179

4.13 Thermal properties of the mine and soil for the natural heating test 183

4.14 Natural heating: Reduction in the number of nodes using non-uniform grids. . . . 185

4.15 Comparison of the speedups using different FPGAs 186

Resumen

Introducción

De acuerdo con informes oficiales, hay más de 100 millones de minas enterradas en todo el mundo y, aunque éste es un problema que viene de lejos, ha alcanzado una especial relevancia con los conflictos de Bosnia en 1995 y de Afganistán en 2001. Al finalizar los conflictos armados, las minas permanecen enterradas y causan entre 15.000 y 20.000 víctimas civiles al año en 90 países, (ICBL, 2006). Teniendo en cuenta que cada año se encuentran y destruyen 100.000 minas, (Horowitz, 1996), se necesitarían 1000 años para eliminar todas las minas que se encuentran diseminadas por el mundo; sin embargo, cada año se entierran 1.9 millones de minas. Se ve así la necesidad de desarrollar nuevas técnicas que permitan llevar a cabo la detección con rapidez y precisión.

Producir una mina es un proceso muy barato (3 $), pero sus efectos son devastadores en las regiones afectadas, (Anderson and Sousa, 1995). La presencia de minas enterradas no tiene sólo implicaciones humanitarias, si no que también tiene implicaciones medioambientales y económicas.

En la actualidad existen más de 350 tipos de minas, fabricadas en más de 50 países, (Vines and Thompson, 1999), sin embargo pueden ser clasificadas en dos grandes categorías:

- Minas antipersona.
- Minas antitanque.

La función de ambos tipos de minas es la misma; sin embargo, existen diferencias entre ellas. Las minas antitanque son bastante grandes y pesadas (2-5 Kg) y suelen estar dispuestas

formando patrones regulares. Este tipo de minas contienen también gran cantidad de explosivo, ya que su propósito es destruir tanques o vehículos. Por otro lado, las minas antipersona contienen menos cantidad de explosivo y son más ligeras. Este tipo de minas se suelen disponer formando patrones aleatorios con el fin de dificultar su localización. Las minas antipersona están diseñadas para herir o matar personas y se activan mediante la presión. El problema de estas minas es que al ser ligeras, la lluvia puede modificar su posición, con lo que los posibles mapas de colocación de minas dejan de ser válidos. Como se puede deducir, la detección de las minas antipersona presenta más dificultades que la detección de las minas antitanque.

Nos centraremos en la detección de minas antipersona, donde las técnicas clásicas de evaluación no destructiva (NDE) fallan. La razón es que contienen poca cantidad de explosivo, son pequeñas y ligeras, y su contenido de metal es pequeño o nulo. No hay una técnica que funcione bien en todas las situaciones (tipo de mina, terreno ...). Entre las técnicas que podemos considerar clásicas están el uso de detectores de metal o magnetómetros. Estos últimos se usan para detectar objetos ferromagnéticos midiendo la perturbación del campo electromagnético natural de la Tierra. Sin embargo, muchas minas modernas no tienen partes metálicas o son muy pequeñas. Aunque en los detectores de metal se puede aumentar la sensibilidad para detectar pequeñas cantidades, este aumento incrementa notablemente el número de falsas alarmas, con lo que no es una solución válida. Así, las técnicas clásicas de detección no son válidas a la hora de detectar este tipo de minas. En los últimos años se han desarrollado diversos tipo de técnicas de detección. Entre ellas podemos citar técnicas que usan la acústica, (Sabatier and Xiang, 2001); Rayos-X, (Lockwood et al., 1997); biosensores, (Larsson and Abrahamsson, 1993); espectrómetros de movilidad de iones, (Jankowski et al., 1992); resonancia nuclear con cuadripolos, (Englelbeen, 1998); análisis de neutrones, (Bach et al., 1996); termografía de infrarrojos, (López et al., 2004; Sendur and Baertlein, 2000; Maksymonko and Le, 1999); radar, (Kosmas et al., 2002; Savelyev et al., 2007).

El radar y la termografía de infrarrojo son dos de las técnicas que más se desarrollan en la actualidad. El radar está basado en la emisión de ondas electromagnéticas en el suelo y en el estudio de la respuesta de éste. Por otro lado, la termografía de infrarrojo se basa en la evolución térmica del suelo ante procesos de calentamiento y/o enfriamiento. En esta última técnica las cámaras de infrarrojo permiten medir la temperatura en la superficie. La presencia de una mina provoca una perturbación en el suelo en ambos casos. Aunque el uso del radar es una técnica más madura, el uso de la termografía ofrece buenos resultados en la detección de minas enterradas a poca profundidad (10-15 cm), que es justo donde el

radar falla debido a la reflexión que se produce en la interfaz aire-suelo.

Evaluación no destructiva de suelos usando termografía de infrarrojos

Los sensores de infrarrojo son sensibles a un determinado rango del espectro electromagnético (infrarrojo). La radiación que reciben los sensores puede ser de origen natural (radiación solar reflejada por el suelo) o artificial (p.ej. iluminación con láser), lo que lleva a definir la termografía pasiva y la activa. No centramos en la termografía pasiva, que está basada en los procesos de calentamiento/enfriamiento que produce la radiación solar en el suelo. Debido a que cada material tiene una respuesta característica a un estímulo térmico, denominada *signatura térmica*, los procesos de enfriamiento/calentamiento afectan de forma diferente a los objetos enterrados y al suelo. Esto se debe a que la mina es un mejor aislante térmico que el suelo; así, durante el día, el suelo situado encima de la mina tiende a acumular más energía térmica porque la mina bloquea el flujo de calor hacia zonas más profundas del suelo. Como resultado de este proceso el suelo sobre la mina tiende a estar más caliente que el suelo que la rodea. Durante la tarde, el suelo situado en la parte superior de la mina pierde energía más rápidamente que el suelo que la rodea y está más frío. Alrededor del mediodía el suelo y la mina alcanzan un equilibrio térmico, con lo cual es imposible llevar a cabo la detección en esa franja horaria, (López et al., 2004). La mayor limitación de esta técnica es que las diferencias de temperaturas están muy influenciadas por las condiciones atmosféricas. Por otro lado, la termografía activa, que usa iluminación artificial, evita esa dependencia de las condiciones atmosféricas.

El uso de la termografía térmica como herramienta de evaluación no destructiva consiste en someter el suelo a procesos de calentamiento/enfriamiento y estudiar su respuesta analizando el patrón térmico obtenido mediante las imágenes de infrarrojo. Una forma eficiente de extraer información a partir de esos datos usando un modelo térmico 3D del suelo basado en la ecuación del calor fue presentado en (López, 2003; López et al., 2004). Las ecuaciones del modelo térmico se resuelven mediante el método de las diferencias finitas. El proceso está dividido en dos partes. En la primera parte el suelo se somete a un proceso de calentamiento y se realiza una comparación entre las temperaturas medidas en la superficie del suelo (a partir de las imágenes de infrarrojo) y las obtenidas mediante simulación asumiendo que no hay minas presentes en el suelo (proceso directo). Las diferencias entre

las temperaturas medidas y las simuladas dan evidencias de la presencia de objetos enterrados en el suelo. En un segundo paso se realiza un proceso de ingeniería inversa, donde se simula el modelo térmico para distintas configuraciones del suelo que representan distintos tipos de objetos enterrados (mina, piedra, ...) y distintas profundidades a las cuales se haya el objeto. La configuración mas cercana a los datos medidos nos permite estimar la naturaleza y posición del objeto. En la Figura 1 se puede ver una representación de todo el proceso de detección. Este método, y particularmente el proceso de ingeniería inversa, que hace un uso intensivo del modelo térmico 3D, supone una carga computacional muy grande para un PC. Para ilustrar esta carga computacional podemos considerar el análisis de una porción de terreno de dimensiones 1m×1m. La aplicación de esta técnica está restringida a una profundidad de 10-15 cm, pero la profundidad de análisis debe ser superior para aplicar las condiciones de contorno (40-50 cm). Usando un grid uniforme con discretización espacial $\Delta x = \Delta y = \Delta z = 0.8$ cm y discretización temporal $\Delta t = 6.25$ s, la simulación de 1 hora del comportamiento térmico del terreno usando C++ en un Pentium IV 3 GHz tarda 5 minutos si empleamos precisión simple para representar las temperaturas (20 minutos si usamos doble precisión). Teniendo en cuenta que el proceso inverso requiere la resolución del proceso directo para múltiples configuraciones del suelo, el tiempo total de cálculo, asumiendo que sólo son necesarias 100 iteraciones (estimación conservativa), llega a las 8 horas (32 hora si usamos doble precisión). Este tiempo es excesivo para aplicaciones que vayan a ser usadas sobre el terreno. De ahí, que sea necesario desarrollar técnicas que permitan reducir el tiempo de cálculo.

En este trabajo exploramos 2 vías para resolver este problema:

- Desarrollo de un sistema hardware que permita resolver las ecuaciones del modelo térmico del suelo mediante el método de diferencias finitas.
- Nuestro objetivo es detectar minas antipersona que están en zonas próximas a la superficie. Además, el límite de la validez de la técnica de termografía de infrarrojo se establece en torno a 10-15 cm de profundidad. Para poder reflejar de forma adecuada el pequeño tamaño de la mina se necesita una gran resolución espacial en el proceso de discretización del modelo en zonas próximas a la superficie. Sin embargo, en capas más profundas será posible incrementar el intervalo de cuantización espacial sin afectar de forma significativa a la temperatura en superficie.

El sistema hardware permitirá acelerar el cálculo proyectando el modelo térmico en una FPGA que actúe como una tarjeta aceleradora. Así, la Figura 2 representa el sistema una

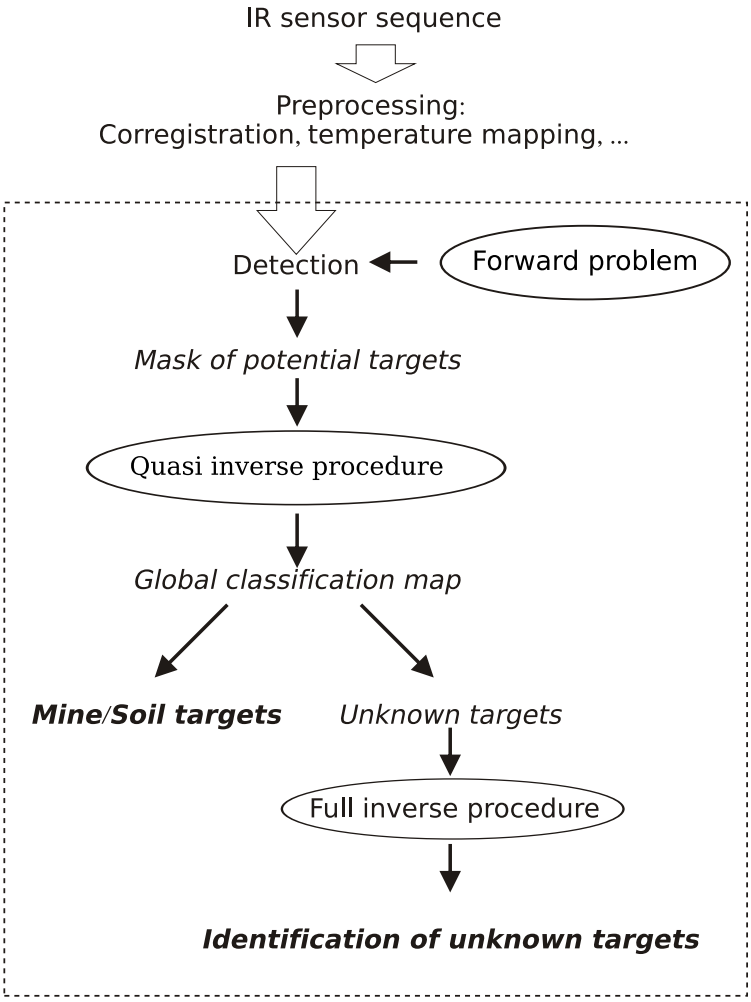


Figura 1: Proceso de detección de minas usando el modelo térmico del suelo.

vez introducida la FPGA, donde los procesos que ésta ejecuta están indicados con líneas discontinuas. Por otro lado, el uso de grids no uniformes permitirá reducir el número de nodos que hay que computar, lo que también permitirá reducir el tiempo de cálculo. Debido a este factor el uso de esta técnica sobre el terreno se hace inviable; así, en este trabajo hemos desarrollado una del modelo térmico sobre una FPGA con el fin de acelerar el proceso de detección.

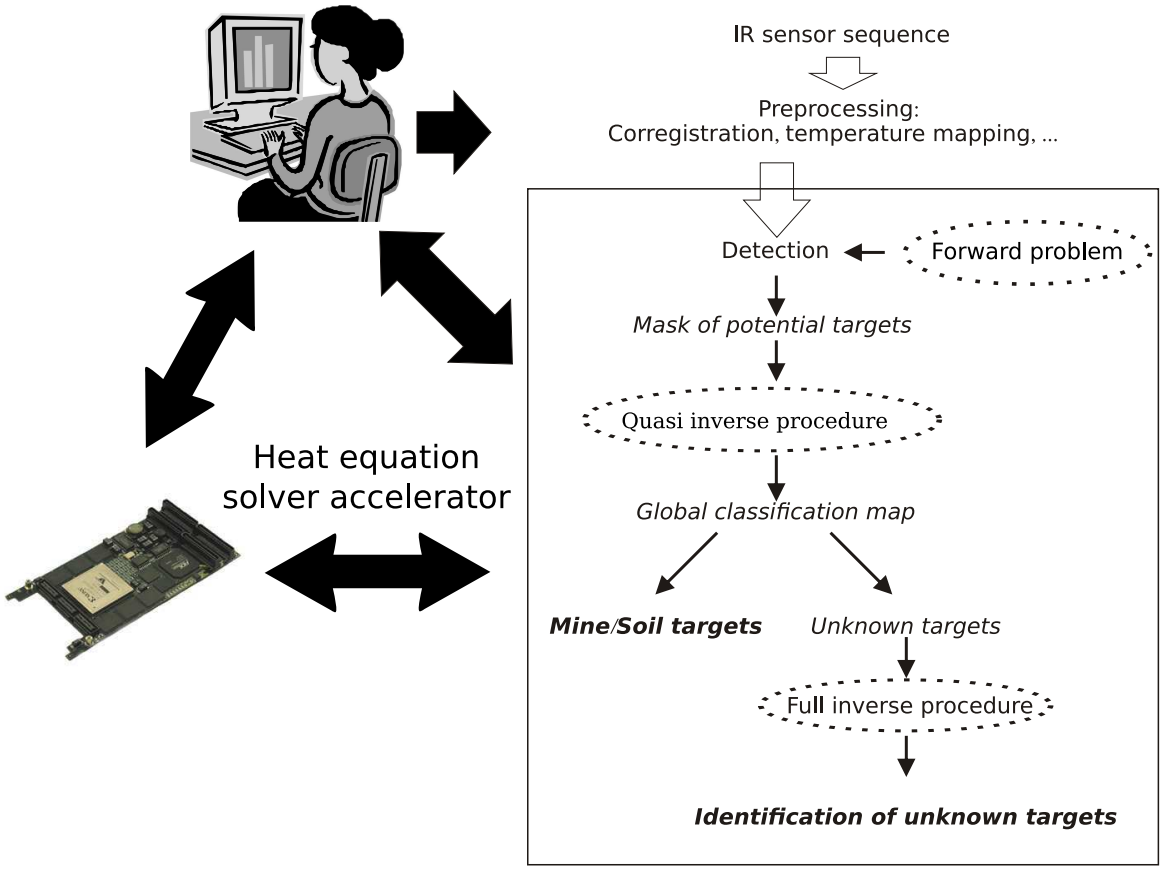


Figura 2: Proceso de detección de minas usando el modelo térmico del suelo usando la FPGA para acelerar el proceso.

Modelo térmico del suelo

En esta sección introducimos brevemente el modelo térmico del suelo usado en la detección de minas antipersona. La ecuación que describe el comportamiento térmico de un objeto es la ecuación del calor, (Bejan, 1993):

$$\frac{\partial T(\overrightarrow{r},t)}{\partial t} - \alpha \nabla^2 T(\overrightarrow{r},t) = 0 \;\; \text{con } \alpha = \frac{k}{\rho c_p} \tag{1}$$

donde $\overrightarrow{r} = (x, y, z)$, ρ es la densidad, c_p es el calor específico, k es la conductividad térmica, α es la difusividad térmica, y T es la distribución de temperaturas en el suelo. Para resolver

esta ecuación se necesitan unas condiciones de contorno y una condición inicial:

$$-k\frac{\partial T(\overrightarrow{r},t)}{\partial n} = q_{net}(t) \quad para\ \Gamma \tag{2}$$

$$\frac{\partial T(\overrightarrow{r},t)}{\partial n} = 0 \quad para\ \partial\Omega\backslash\Gamma \tag{3}$$

$$T(\overrightarrow{r},t=t_0) = T_0(\mathbf{r}) \quad en\ \ \Omega \tag{4}$$

$$T(x,y,z\rightarrow\infty,t) = T_\infty \tag{5}$$

donde q_{net} es el flujo de calor neto en la interfaz aire-suelo, Γ; n es la normal a la superficie considerada, $\partial\Omega$. La ecuación (2) proporciona las condiciones de contorno en la interfaz aire-suelo; la ecuación (3) muestra las condiciones de contorno para el resto de las caras del volumen de suelo considerado, imponiendo un flujo nulo de calor a través de ellas (válido para volúmenes suficientemente grandes); la ecuación (4) establece las condiciones iniciales de temperatura. Finalmente, la ecuación (5) establece que para una profundidad suficiente la temperatura no se ve afectada por los procesos que ocurren en la superficie. En ésta se consideran procesos de convección y radiativos; así, el flujo neto de calor en la superficie se puede escribir como:

$$q_{net}(t) = q_{sun}(t) + q_{rad}(t) + q_{conv}(t) \tag{6}$$

donde q_{sun} es la radiación de baja longitud de onda emitida por el sol, que es una función del albedo del suelo ϵ_{sun}, la declinación solar δ y de la latitud del lugar γ; q_{conv} es el flujo de calor intercambiado debido al proceso de convección que tiene lugar en la superficie, y q_{rad} es el flujo de calor producido por los procesos radiativos.

El término q_{rad} se puede expresar como:

$$q_{rad}(t) = q_{sky}(t) - q_{soil}(t) \tag{7}$$

donde $q_{sky}(t)$ es la radiación emitida por la atmósfera, dada por la Ley de Stephahn-Boltzmann, $q_{sky}(t) = \sigma\epsilon T_{air}^4$, donde T_{air} es la temperatura del aire. La pérdida de calor del suelo debido a procesos radiativos se puede obtener como $q_{soil} = \sigma\epsilon T_{soil}^4$, donde ϵ es la emisividad del suelo y $T_{soil} = T(x,y,0,t)$ es la temperatura del suelo en la superficie. Este es un término no lineal, pero puede ser linearizado como, (Watson, 1973):

$$q_{rad}(t) = \sigma\epsilon T_{sky}^4 - \sigma\epsilon T_{soil}^4 \simeq \sigma\epsilon T_{sky}^3(T_{sky} - T_{soil}) \quad for\ \frac{T_{sky}-T_{soil}}{T_{sky}} << 1 \tag{8}$$

El último término de la ecuación (6) hace referencia al proceso de convección que tiene lugar en la superficie. Este término puede ser aproximado como, (Khanafer and Vafai,

2002):

$$q_{conv}(t) = h(T_{air}(t) - T_{soil}(t)) \tag{9}$$

donde h es el coeficiente de convección.

La ecuación de calor puede ser resuelta empleando métodos numéricos. Nos centraremos en el uso de método de las diferencias finitas en el tiempo (FD-TD), (Incropera and DeWitt, 2002; Bejan, 1993), que será introducido en la siguiente sección.

Simulador FD-TD del modelo térmico

El método FD-TD permite resolver numéricamente ecuaciones diferenciales; así, un medio continuo es reemplazado por un grid de puntos discretos (nodos) y los cálculos ya no se realizan de forma continua si no que se realizan en instantes de tiempo discretos. Con este método la ecuación del calor (1) se transforma en un conjunto de ecuaciones algebraicas que permiten, aplicando un proceso iterativo, obtener el comportamiento térmico del terreno.

Una aproximación para la derivada con respecto al tiempo en un punto (i, j, k) del grid es:

$$\frac{\partial T_{i,j,k}}{\partial t} \simeq \frac{T_{i,j,k}^{m+1} - T_{i,j,k}^{m}}{\Delta t} \tag{10}$$

donde el superíndice m hace referencia a los instantes de tiempo en los que se realizan los cálculos; Δt es el paso de discretización temporal, es decir, el intervalo de tiempo entre dos iteraciones; y los subíndices (i, j, k) identifican un punto del grid. De una forma similar la segunda derivada de la temperatura con respecto a la posición puede ser expresada como:

$$\frac{\partial^2 T(\overrightarrow{r}, t)}{\partial x^2} \simeq \frac{T_{i+1,j,k} + T_{i-1,j,k} - 2\,T_{i,j,k}}{\Delta x^2} \tag{11}$$

$$\frac{\partial^2 T(\overrightarrow{r}, t)}{\partial y^2} \simeq \frac{T_{i,j+1,k} + T_{i,j-1,k} - 2\,T_{i,j,k}}{\Delta y^2} \tag{12}$$

$$\frac{\partial^2 T(\overrightarrow{r}, t)}{\partial z^2} \simeq \frac{T_{i,j,k+1} + T_{i,j,k-1} - 2\,T_{i,j,k}}{\Delta z^2} \tag{13}$$

donde Δx, Δy y Δz son los pasos de discretización espacial en las direcciones x, y y z. Por lo tanto los cálculos están restringidos a puntos discretos tanto en el espacio como en el tiempo. Una vez aplicada la discretización, tanto espacial como temporal, a la ecuación (1),

obtenemos para un nodo de la superficie:

$$\begin{aligned} T^{m+1}_{i,j,1} =& (1-6F_0)T^m_{i,j,1} + 2\alpha_{sun}F_0 Sq^m_{sun} + 2F_0 H(T_{air} - T^m_{i,j,1}) + \\ & F_0(T^m_{i+1,j,1} + T^m_{i-1,j,1} + T^m_{i,j+1,1} + T^m_{i,j-1,1} + 2T^m_{i,j,2}) + \\ & 8F_0 R T^3_{sky}(T_{sky} - T^m_{i,j,1}) \end{aligned} \tag{14}$$

donde hemos asumido, sin pérdida de generalidad, que $\Delta x = \Delta y = \Delta z$, y donde:

$$F_0 = \frac{\alpha \Delta t}{(\Delta x)^2} \qquad R = \frac{\sigma \epsilon}{k}\Delta x \qquad S = \frac{\Delta x}{k} \qquad H = hS \tag{15}$$

Para un nodo interno se obtiene:

$$T^{m+1}_{i,j,k} = (1-6F_0)T^m_{i,j,k} + F_0(T^m_{i+1,j,k} + T^m_{i-1,j,k} + T^m_{i,j+1,k} + T^m_{i,j-1,k} + T^m_{i,j,k+1} + T^m_{i,j,k-1}) \tag{16}$$

Se obtiene así un conjunto de ecuaciones que proporcionan la evolución de la temperatura de los nodos del grid. Para obtener el comportamiento térmico del suelo este conjunto de ecuaciones se ejecuta de forma iterativa hasta que se alcance el tiempo de simulación deseado. Este método de resolución numérica de ecuaciones diferenciales puede presentar inestabilidad si los términos de discretización temporal y espacial no son convenientemente elegidos. Así, para que el sistema sea estable se debe cumplir que el término que multiplica a $T^m_{i,j,k}$ en la Ecuación (16) sea mayor que creo, lo que se refleja en:

$$1 - 6F_0 > 0 \implies F_0 = \frac{\alpha \Delta t}{(\Delta x)^2} < \frac{1}{6} \tag{17}$$

Así, vemos que hay una relación entre la resolución espacial y temporal que debemos tener en cuenta a la hora de realizar la discretización del la ecuación del calor.

Implementación hardware del modelo térmico

El sistema que simula el comportamiento térmico del suelo está compuesto de una parte software y una parte hardware, ver Figura 3. La parte software ha sido desarrollada usando C++ y es ejecutada en un PC (HOST), mientras que la parte hardware ha sido desarrollada usando Handel-C y VHDL, (Celoxica, 2005a; VHDL, 1987), y se ha implementado en una placa RC2000 que consta de una FPGA Virtex-II, 8 bancos de memoria y una tarjeta portadora PCI para que pueda ser conectada a un PC. Se ha optado por una solución mixta VHDL-Handel-C, donde VHDL se empleó para desarrollar el núcleo de procesamiento y Handel-C para controlar las comunicaciones con la parte software y con los recursos

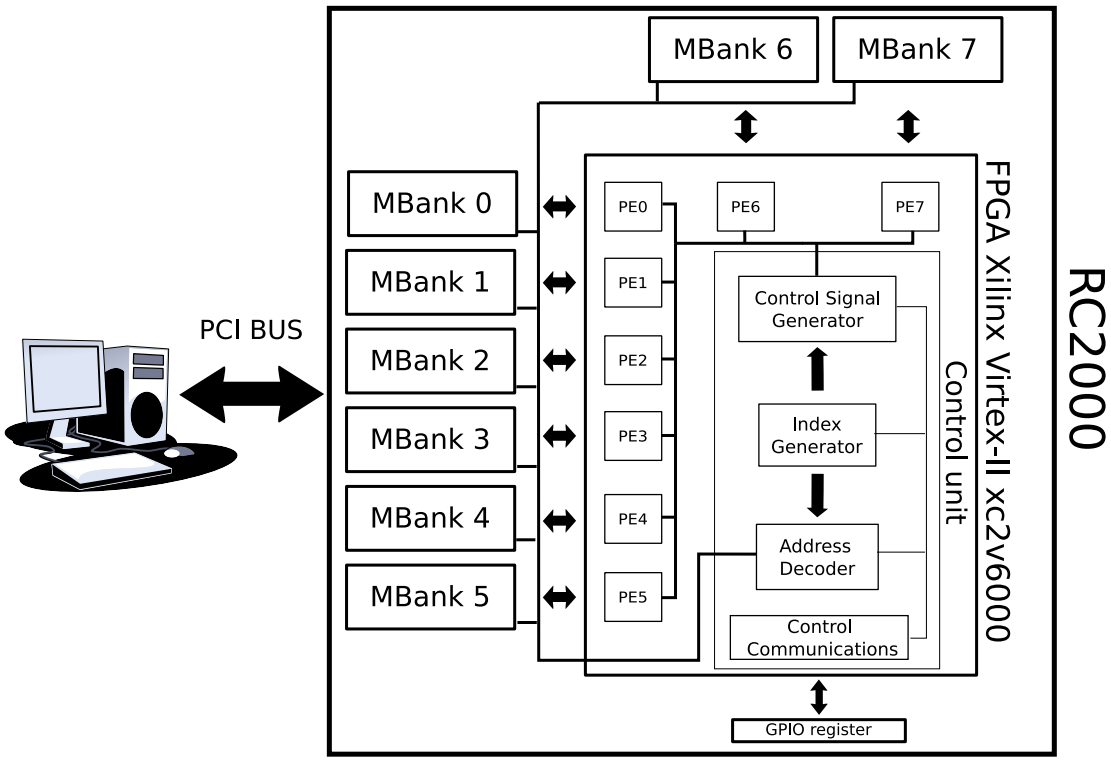


Figura 3: Sistema que implementa el modelo térmico del suelo.

periféricos, ya que los lenguajes clásicos de descripción de hardware como VHDL permiten obtener sistemas con un mayor rendimiento comparados con lenguajes de descripción de hardware de alto nivel, como puede ser Handel-C, (Sullivan and Saini, 2003; Ortigosa et al., 2006). Describimos ahora brevemente las distintas partes que componen el sistema.

HOST

En esta arquitectura el HOST carga en los bancos de memoria de la RC2000 y en la select-RAM de la FPGA los datos necesario para iniciar el procesamiento: las temperaturas iniciales de todos los nodos del grid, los parámetros térmicos del suelo, las condiciones de contorno y los parámetros de la simulación (número de nodos, número de iteraciones, ...). Usando este esquema evitamos hacer el Place&Route cada vez que se quiera hacer una simulación con unos parámetros distintos.

Hardware

A la hora de diseñar la parte hardware hay que tener en cuenta que debido al gran número de nodos involucrados en las simulaciones (cientos de miles), un mapeado nodo - elemento de procesamiento está fuera de las posibilidades de cualquier FPGA o incluso de un ASIC. Como se puede ver en la Figura 3, el sistema está compuesto de 8 elementos de procesamiento (PE) y una unidad de control que genera tanto las direcciones de memoria de los datos que deben ser leídos, como las señales de control necesarias para el correcto funcionamiento del sistema. A pesar de que la utilización de la FPGA es baja, sólo se han implementado 8 PEs debido al acceso a los bancos de memoria, que limitan el número de temperaturas que pueden ser leídas en paralelo, y por tanto, el número de PE que pueden estar continuamente recibiendo datos.

Los 8 bancos de memoria de la RC2000 se usan para almacenar los valores de las temperaturas, donde los datos están distribuidos en z-capas y dos capas consecutivas son almacenadas en dos bancos consecutivos; así, el banco de memoria 0 almacena la capa 0 (z=0, superficie), la capa 8 (z=8) y así sucesivamente. Todos los bancos de memoria están divididos en dos mitades idénticas, donde una mitad se usa para leer las temperaturas y la otra para almacenar los valores de temperatura actualizados. Una vez que se ha completado una iteración, los roles de las dos mitades se intercambian.

Cada uno de los PE sigue la estructura de las ecuaciones (14), (16), y por lo tanto, está formado por sumadores/restadores, multiplicadores y elementos de memoria. En el elemento de procesamiento se pueden distinguir 4 bloques funcionales: un bloque de entrada, denominado $INPUT$, dos bloques de sumadores/restadores denominados SUB y ACC y una etapa de salida denominada $Mult - Output$. $PE0$ y $PE7$ presentan unas pequeñas diferencias en la etapa de entrada debido a cómo reciben estos dos PE los datos. Además, $PE0$ incluye una circuitería adicional que computa las condiciones de contorno cuando se está actualizando la temperatura de un nodo superficial.

Resultados

Dentro del trabajo realizado podemos distinguir dos contribuciones principales:

- Implementación en una FPGA del modelo térmico.

- Estudio de la reducción del número de nodos aplicando discretizaciones no uniformes en profundidad.

Con la implementación hardware del modelo térmico incrementamos la velocidad de simulación, permitiendo el uso de esta técnica en situaciones reales. La implementación hardware consigue reducir en un factor 34 el tiempo de cálculo con respecto a un Pentium IV 3 GHz. Por otro lado, se ha propuesto un modelo en el que la discretización no es uniforme en profundidad, incrementando el espaciado entre capas para profundidades grandes. Esta nueva discretización del modelo permite reducir en un 30% el número de nodos del grid sin que afecte de forma importante a la distribución de temperaturas obtenidas en superficie. Así, se ha validado tanto la implementación hardware, realizada en aritmética entera, como el uso de grids no uniformes obteniendo diferencias con respecto al modelo original que no afectan a la *performance* del sistema. También hemos contemplado la posibilidad de usar FPGAs más potentes que permitan reducir tiempos de cálculo. Así, hemos explorado el uso de una FPGA Virtex-5 de Xilinx, obteniendo un factor de aceleración teórico de 210 comparado con la solución software.

Summary

Non-destructive Evaluation (NDE) is an interdisciplinary field of research integrating advances in measurement, analysis and information processing techniques for the quantitative characterization of materials and structures by non-invasive means. Applications are ubiquitous in fields such as medical diagnosis, clearance of minefields and on-line manufacturing process control, as well as the traditional NDE areas of flaw detection and materials characterization. Despite the broad range of applications, all NDE tests are characterized by the use of some form of energy (for instance, electromagnetic, ultrasonic, radiographic or thermographic) injected into the inspected material and the evaluation of the interaction between the energy and the material.

In this work we are concerned with the non destructive evaluation of soils for the detection of buried landmines, particularly small plastic antipersonnel mines which are virtually undetectable by traditional methods. To this aim several techniques such as Ground Penetrating Radar (GPR) or infrared thermography (IRT) have been considered. The main drawback of GPR is the difficulty of detecting shallowly buried objects due to the large reflection coming form the air-ground interface, which tends to mask reflections from objects just underneath the surface. On the other hand, infrared thermography has been shown as an efficient technique to detect buried objects at a maximum depth of 10-15 cm. As antipersonnel mines are usually buried near the surface, we have chosen the infrared thermography as detection technique. Its use consists on subjecting the area under inspection to a source of natural or artificial heating/cooling process and studying the soil's response by means of the analysis of its thermal evolution given by a temporal sequence of infrared images. A satisfactory way of extracting information regarding the presence and exact location of the mines from such data based on the solution of the heat equation was presented in (López, 2003; López et al., 2004). The process is divided in two steps. On the first one the soil is subjected to a heating process and a comparison between temperatures measured at the soil surface (through IR imaging) and those obtained by simulation using

the model under the assumption of homogeneous soil and mine absence is made, the so called forward problem. The differences between measured and simulated data put into evidence the presence of unexpected objects on the soil. The second step is an inverse engineering problem where the thermal model must be run for multiple soil configurations representing different types of possible targets (mine, stone, ...) and depths of burial. The nearest configuration to the measured data gives us the estimated nature and location of the targets. This approach and, particularly, the inverse engineering process, makes an intensive use of the 3D thermal model that needs to be solved iteratively involving complex, coupled sets of partial differential equations. The extensive computing power required makes its software implementation impractical. The challenge in this case is on the efficient solution of the aforementioned model. To this aim, the Finite-Difference Time-Domain (FD-TD) method has been used in order to express the set of equations describing the model in a discrete way. FD-TD has been successfully used in a variety of fields, most notably on electromagnetic simulations, but its applicability has been traditionally limited due to its demanding requirements in terms of memory and computing power. This limitation can be, however, avoided by means of a hardware implementation.

To illustrate the required computing power of the detection algorithm previously described, we will consider the analysis of a piece of soil of moderate dimensions, of 1 m×1 m. As explained, the scope of applicability of IR techniques for mine detection is restricted to a depth of barely 10-15 cm, but the depth of analysis must be set to at least 40-50 cm in order assume a constant temperature at such a depth. Using a uniform spatial discretization of $\Delta x = \Delta y = \Delta z = 0.8$ cm and assuming a temporal discretization step of $\Delta t = 6.25$ s, the simulation of the behavior of the soil during one hour using C++ on a Pentium IV 3GHz takes 5 minutes if single precision arithmetic is used to represent the temperatures (20 minutes if double precision is used). Taking into account that the proposed inverse procedures require the solution of the forward model for multiple soil configurations, the total computing assuming that only 100 iterations are needed (a soft approach) will add up to 8 hours (32 hours in double precision). As this time is excessive for on the field experiments arises the need to reduce it.

The aim of this work was to develop efficient techniques to reduce the computing time of the detection algorithm working in two different ways:

- Development of a hardware solver for the thermal model.
- Using non-uniform grids in the discretization scheme to reduce the number of nodes.

We present a fully 3D FD-TD hardware solver of the heat equation applied, but not limited, to the thermal model simulation of the soil for the detection of buried landmines, with a significant speedup over a PC, using a commercial FPGA platform from Celoxica. The system was designed using VHDL and Handel-C; the processing core was developed using VHDL, whereas Handel-C was only used to perform the communications with the outside of the FPGA. Additionally, the use of non-uniform grids allows to reduce the number of layers of the thermal model, thereby reducing the computing time.

The thesis is outlined as follows. In Chapter 1 a review of the main NDE techniques is done and we will also introduce the main NDE techniques used to detect buried landmines, centering our attention on the use of infrared thermography and ground penetrating radar. In this chapter we will also introduce the physical thermal model and a brief summary of the detection algorithm proposed in (López, 2003). In Chapter 2 the applications where the FD-TD method has been used are introduced; we will also make a review of the FD-TD hardware implementations. Moreover, this chapter introduces the FD-TD algebraic equations for different situations: interface between two different materials, use of non-uniform grids and a combination of both situations. Chapter 3 addresses the hardware implementation of the thermal model. A detailed review of the system architecture and all aspects related to the implementation is made, including the description of the processing elements and the complete data flow of the system. In Chapter 4 the results are shown, where several simulations has been carried out to test the validity of the hardware implementation and the use of non-uniform grids. The hardware implementation speed ups the computations by a factor of 34 compared to a software solution, while with the use of non-uniform grids the nodes that need to be computed are reduced in a 30%. Finally, the main conclusions and future work are presented in the last chapter.

Chapter 1

Non-destructive evaluation

1.1 Introduction

Non-destructive evaluation (NDE) or non-destructive testing (NDT) is a set of specialized technical inspection methods which provide information about the condition of materials and components without impairing its future use. The art and science of NDE are very old and, probably, one of the most famous and well known examples is that of Archimedes, who discovered the principle that has his name analyzing the composite of a crown. The bigger development in NDE took place during World War II. The progress in materials engineering, identifying new materials and improving the existing ones after a number of catastrophic failures in World War II like the brittle fracture of Liberty ships, required to test and improve material properties. This resulted in a wider application of the then existing NDE methods and techniques and on the development of new ones. In the beginning, NDE was primarily used for process control and secondarily for quality control and it was soon recognized by the industry as a means of meeting consumer demands for better products, reduced cost and increased production. From an industrial point of view, the purpose of NDE is to determine whether a material or a component will satisfactorily perform its intended function. In general, the purpose of NDE techniques will fall into one of the following categories:

- Determination of material properties.
- Detection, characterization, location and sizing of discontinuities/defects.

- Determination of the quality of manufacture or fabrication of a component/structure.
- Checking for deterioration after a period of service for a component/structure.

By detecting faulty components and thus preventing the loss of material, manpower and time to market, NDE increases productivity, and hence the economic gains. Preventive maintenance tells if parts are still satisfactory for use, which pays off in terms of dependable predictable production, fewer repairs, less accidents and lower over-all operating costs. This results in an increased serviceability of equipment and materials by finding and locating defects which may cause malfunctioning or breakdown of equipment, (Pruitt and Lebsack, 1994). In the field of safety, proper use of NDE will aid in the prevention of accidents, with their possible loss of life, property, and vital equipment. Another application is the identification of materials differing in metallurgical, physical or chemical properties which can often be done by using NDE methods, (Hellier, 2000). NDE techniques can also be used as an aid in new process and manufacturing techniques.

Thus, NDE techniques are widely used in nowadays industry with a lot of applications in the development of new materials, studying material properties or analyzing defects. NDE techniques have medical applications, such as magnetic resonance imaging, radiographies and computed tomography. These techniques are also used for the non-destructive evaluation of soils, such as in demining tasks. In this Chapter a brief review of the most significant NDE techniques and their areas of application is made.

1.2 NDE techniques

A variety of NDE techniques have been developed to detect and characterize materials and defects. All these techniques are based on physical principles and nearly every property of the materials to be inspected has been made the basis for some method or technique of NDE. Most NDE methods involve subjecting the material under examination to some source of external energy and analyzing the response signals. The essential parts of any NDE test are:

- Energy source used to probe the test object (such as X-rays, ultrasonic waves or thermal radiation).

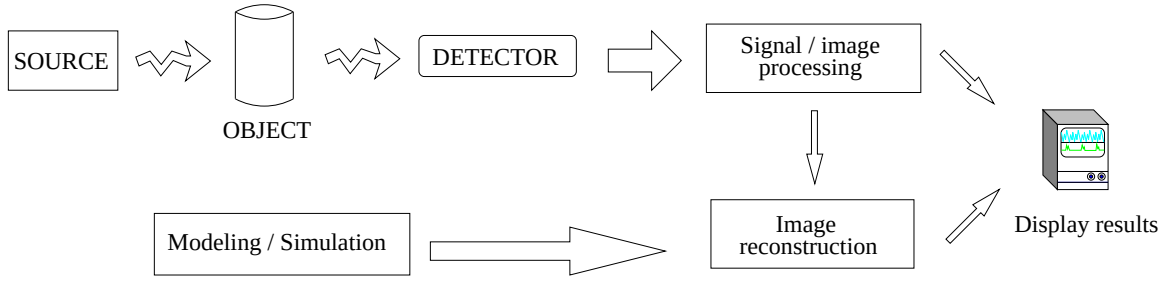


Figure 1.1: Different elements of a NDE system.

- Image or signature resulting from interaction with the test object (e.g. attenuation of X-rays or reflection of ultrasound).
- A detector to acquire information from the object under study.
- In some cases, a manipulator to translate, elevate or rotate the object.
- A computer for control, data acquisition, processing and analysis.

In most techniques, NDE results are indirect measurements. Hence, it is essential that the interpretation is made by an skilled person. The expert interpreting the results sometimes determines the success or failure of a test method or technique. In Fig. 1.1 a scheme of a NDE system can be seen, where we can distinguish the aforementioned parts of a generic NDE system. In this figure there is also a modeling or simulation component, that is not always present in all NDE techniques. The modeling is introduced when the response of the object to the energy source is predictable and the result from the simulation is compared with the real response of the object to the source, hence detecting defects or anomalies in the analyzed object.

There is a broad spectrum of NDE techniques and their goals can be divided into:

- Measurement of discontinuities (such as cracks, voids, inclusions, delaminations).
- Obtention of the structure or malstructure (including crystalline structure, grain size, segregation, misalignment).
- Measurement of dimensions (thickness, diameter, gap size, discontinuity size).
- Obtention of the value of physical and mechanical properties (reflectivity, conductivity, elastic modulus, sonic velocity).

- Analysis of the composition (alloy identification, impurities, elemental distributions).
- Measurement of the stress and dynamic response (residual stress, crack growth, wear, vibration).
- Performance of a signature analysis (image content, frequency spectrum, field configuration).

In the following sections we show a brief review of the nowadays most widely used NDE techniques.

1.2.1 Visual inspection

Visual inspection (VI) techniques are widely used to ensure product reliability during manufacturing and to examine any gross discrepancies on the surface of operating components. This method requires good vision, good lighting and an a priori knowledge of what to look for. Visual inspection can be enhanced by various methods ranging from low power magnifying glasses to boroscopes. These techniques involve illumination of object surface with light and examination of the reflected light using visual aids, usually at magnification, (McIntire and Moore, 1996). Visual examination can reveal gross surface defects, cleanliness, foreign objects, surface condition, mismatches and any other discrepancies, (Baldev et al., 2002). VI techniques are, in fact, considered to be the most effective and the least expensive NDE technique because it can easily carried out and usually does not requires special equipment. In some cases surface preparations must be made prior to the examination; this previous step can range from wiping with a cloth to blast cleaning and treatment with chemicals to show the surface details. This technique can sometimes identify where a failure is most likely to occur and identify when a failure has commenced. In general, VI is often enhanced by other surface methods of inspection, which can identify defects that are not easily seen by the eye.

1.2.2 X and gamma ray imaging

X-rays and gamma rays are electromagnetic radiation of a much shorter wavelength ($\sim 10^{-4}$ Å) than visible light ($\sim$ 600 nm). This very short wavelength is what gives X-rays and gamma rays their power to penetrate materials that visible light can not. High voltage

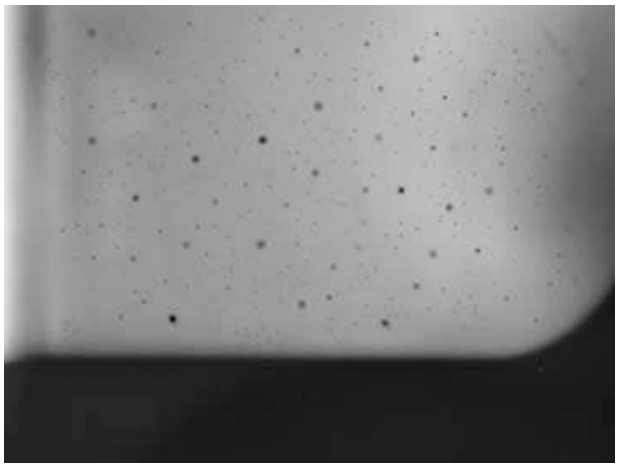


Figure 1.2: X-ray image showing holes caused by gas accumulation on a metal piece.

machines produce X-rays whereas gamma rays are produced from radioactive isotopes such as iridium-192.

The choice of which type of radiation is used largely depends on the thickness of the material to be tested and the ease of access to the area of inspection. Gamma sources have the advantage of portability, which makes them ideal for use in construction site working. Both X-rays and gamma rays are very hazardous and special precautions must be taken when performing radiography. Therefore, the method must be undertaken under controlled conditions, inside a protective enclosure or after assessment with appropriate barriers and warning systems in order to ensure that hazards to personnel cannot occur.

The techniques that use these kind of radiations are widely used in industrial, military and medical applications to determinate internal structure of objects, (Levine et al., 1999), and to detect internal defects, such as voids or cracks, (Zscherpel and Alekseychuk, 2003). In Fig. 1.2 an example of an X-ray image can be seen; the image represents a piece of metal that has been been affected by gas accumulation producing holes in its structure. Much of the use of these techniques is due to the development of the computational tomography (CT). CT was first used as a medical diagnostic tool in the 70's but in the mid 80's it was adapted to be used in non-medical or industrial applications. Single view radiography hides information of object features and the depth of these features. CT, on the other hand, gives three dimensional (3D) information about the objects by obtaining several radiographic images acquired at different angles which are then processed in a computer, (Barret and Swindel, 1981) and (Kak and Slaney, 1987). The final 3D image, generated combining all images, provides exact locations of internal and external features of the

object. The main limitation of these techniques is that the objects under study should not be damaged by X-rays or gamma radiation.

1.2.3 Liquid penetrant inspection

Liquid penetration inspection (LPI) is a method used to reveal surface breaking defects by bleed out of a colored or fluorescent dye from the flaw. The technique is based on the ability of a liquid to be drawn into a "clean" surface breaking flaw by capillary action. The first step of this technique is the surface preparation, that must be free of oil, grease, water, or other contaminants that may prevent the penetrant from entering flaws. Once the surface has been cleaned, the penetrant material is applied by spraying, brushing, or immersing the parts in a penetrant bath. The penetrant is left on the surface for a sufficient time, in order to allow as much penetrant as possible to be drawn from or to seep into a defect. Once the penetrant has been applied, the excess penetrant must be removed. This is the most delicate part of the inspection procedure because the excess penetrant must be removed from the surface of the sample while removing as little penetrant as possible from defects. Finally, a thin layer of developer is applied to the sample to draw penetrant trapped in flaws back to the surface where it will be visible. Colored (contrast) penetrants require good white light while fluorescent penetrants need to be used in darkened conditions with an ultraviolet "black light", (Noel and Moore, 1999). In Fig. 1.3(a)-(c) we can see the different steps of the LPI technique. Fig. 1.3(a) shows the application of the liquid to the object being analyzed; Fig. 1.3(b) shows the removal of the liquid and how it is trapped on the flaw. Finally, in Fig. 1.3(c) the developer is applied which produces the flowing of the liquid to the surface making possible the defect detection.

LPI is one of the most widely used NDE methods and its popularity can be attributed to two main factors, which are its relative ease of use and its flexibility. LPI can be used to inspect almost any material provided whose surface is not extremely rough or porous. Materials that are commonly inspected using LPI include metals, glass, ceramic materials, rubber or plastics. LPI offers flexibility in performing inspections because it can be applied in a large variety of applications ranging from automotive spark plugs to critical aircraft components. Summarizing, LPI technique has some advantages:

- Highly sensitive to small surface discontinuities.
- Few material limitations.

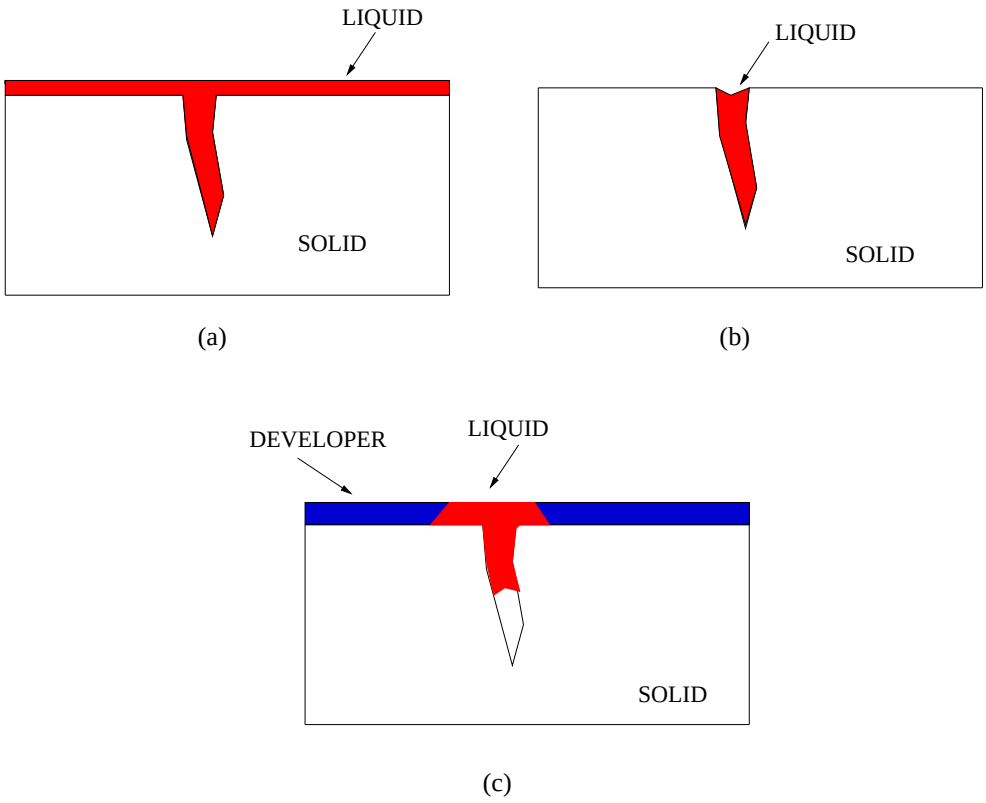


Figure 1.3: Different steps of the Liquid Penetrant Inspection: (a) Penetrant application; (b) penetrant removal and (c) developer application.

- Large areas and large volumes of parts/materials can be inspected rapidly and at low cost.
- Parts with complex geometric shapes are routinely inspected.
- Indications are produced directly on the surface of the part and constitute a visual representation of the flaw.
- Penetrant materials and associated equipment are relatively inexpensive.

However, like all NDE methods, LPI also has some disadvantages:

- Only surface breaking defects can be detected.
- Only materials with a relative non-porous surface can be inspected.
- Precleaning is critical as contaminants can mask defects.

- The inspector must have direct access to the surface being inspected.
- Surface roughness can affect inspection sensitivity.
- Multiple process operations must be performed and controlled.
- Chemical handling and proper disposal is required.

1.2.4 Ultrasonic testing

Ultrasonic testing (UT) uses high frequency sound energy to conduct examinations and make measurements, (Krautkrämer and Krautkrämer, 1990). In this technique the object is radiated with high frequency sound waves ($\sim$ 1 MHz) and the response of the object is detected with an appropriate sensor. The ultrasound waves are modified inside the object, suffering attenuation, reflection and scattering, giving information about the internal structure of the object. The most common application of ultrasound is the detection of defects in metals and the measurement of thickness. A typical UT inspection system consists of several functional units, such as the pulser/receiver, transducer, and display devices. A pulser/receiver is an electronic device that can produce high voltage electrical pulses. Driven by the pulser, the transducer generates high frequency ultrasonic energy. The sound energy is introduced and propagates through the materials in the form of waves. When there is a discontinuity (such as a crack) in the wave path, part of the energy will be reflected back from the flaw surface. The reflected wave signal is transformed into electrical signal by the transducer and is displayed on a screen. From the signal, information about the reflector location, size, orientation and other features can sometimes be obtained. In Fig. 1.4 we can see the behavior of an UT system; as can be seen the defect modifies the ultrasonic impulse in a way different from the rest of the object, where defects are not present.

Ultrasonic inspection is a very useful and versatile NDE method and has among its advantages:

- It is sensitive to both surface and subsurface discontinuities.
- The depth of penetration for flaw detection or measurement is superior to other NDE methods.
- Only single-sided access is needed.

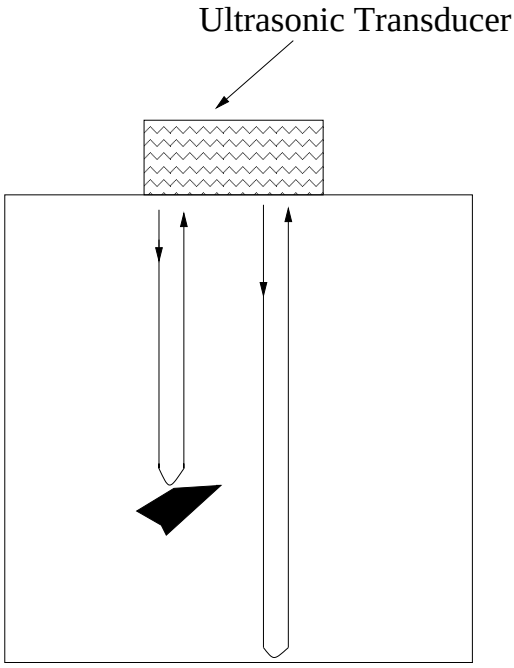


Figure 1.4: UT system to detect flaws.

- It is very accurate in determining reflector position and estimating size and shape.
- Minimal part preparation required.
- Electronic equipment provides instantaneous results.
- Detailed images can be produced with automated systems.
- It has other uses, such as thickness measurements, in addition to flaw detection.

However UT, like all NDE methods, has its limitations, among them we can cite:

- Surface must be accessible to transmit ultrasound.
- It normally requires a coupling medium to promote transfer of sound energy into the test object.
- Materials that are rough, irregular in shape, very small, exceptionally thin or not homogeneous are difficult to inspect.
- Cast iron and other coarse-grained materials are difficult to inspect due to low sound transmission and high signal noise.

- Linear defects oriented parallel to the sound beam may go undetected.

1.2.5 Electromagnetic inspection or eddy current inspection

Eddy current (EC) inspection is one of the several NDE methods that uses electromagnetism as the basis for examinations of conductive materials. Eddy currents are created through a process called electromagnetic induction. When an alternating current is applied to a conductor, such as a copper wire, a magnetic field develops in and around the conductor. This magnetic field expands as the alternating current rises to maximum and collapses as the current is reduced to zero. If another electrical conductor is brought into the close proximity to this changing magnetic field, current will be induced in this second conductor. In EC technique, a coil (also called probe or sensor) is excited with sinusoidal alternating current (frequency 50 Hz - 5 MHz) to induce eddy currents in the electrically conducting material such as stainless steel, aluminum, etc, being tested. The change in coil impedance that arises due to the distortion of eddy currents at regions of discontinuities (defects, material property variations, surface characteristics, etc) and associated magnetic flux linkages, is measured and correlated with the cause producing it, that is, the discontinuities. Summarizing, in EC technique the following sequence happens: first EC coil generates primary magnetic field (Ampere's law); then this primary magnetic field induces eddy currents in the material (Faraday's law); these eddy currents generate a secondary magnetic field in the opposite direction (Lenz's law); thus the coil impedance changes, as a result of this secondary magnetic field; and finally the impedance change is measured, analyzed and correlated with defect dimensions. In Fig. 1.5 the scheme of the EC technique is shown; we can distinguish the coil that is connected to an alternate generator and how it induces the eddy currents on the material that it is being examined.

One of the major advantages of eddy current as an NDE tool is the variety of inspections and measurements that can be performed. Eddy currents can be used for crack detection, material thickness or conductivity measurements. Some of the advantages of eddy current inspection include:

- Sensitive to small cracks and other defects.
- Very shallow and tight surface fatigue cracks and stress corrosion cracks (5 microns width and 50 microns depth) can be detected
- Inspection gives immediate results.

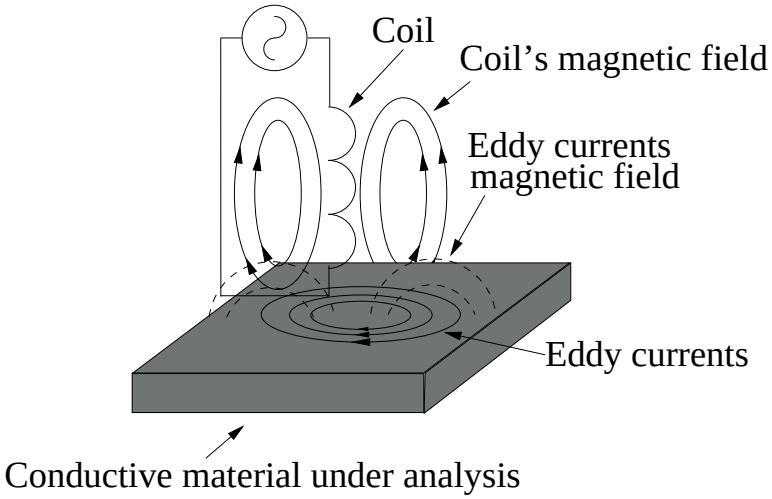


Figure 1.5: Eddy current scheme to evaluate defects.

- Equipment is very portable.
- Test probe does not need to contact the part.
- Inspects complex shapes and sizes of conductive materials.

However the eddy current NDE method also have some limitations that include:

- Only conductive materials can be inspected.
- Surface must be accessible to the probe.
- Skill and training required is more extensive than in other techniques.
- Surface roughness may interfere.
- Depth of penetration is limited.
- Defects such as delaminations that lie parallel to the probe coil winding and probe scan direction are undetectable.

Eddy current inspection is used in a variety of industries to find defects and make measurements. One of the primary uses of eddy current testing is for defect detection when the nature of the defect is well understood. In general, this technique is used to inspect a relatively small area and the probe design and test parameters must be established with a

good understanding of the flaw that is tried to be detected. Since eddy currents tend to concentrate at the surface of a material, they can only be used to detect surface and near surface defects, (Birnbaum and Free, 1981).

1.2.6 Metal detector

Metal detectors are electromagnetic inspection devices and they are based on electromagnetic induction (EMI). These devices are widely used in landmine detection because the metal detector produces a time-varying magnetic field and the metal parts present in the mine produce an eddy current which allows to detect the mine. Thus, the metal detector senses this current detecting then the presence of an object. The frequency of the magnetic field is limited to a few tens of kHz. The primary field is produced by an electrical current flowing in a coil of wire (transmit coil), and the secondary field is usually detected by sensing the voltage induced in the same or another coil of wire (receive coil). Present research is investigating replacement of the receive coil with magnetoresistive devices.

This technology is very mature and has been used for landmine detection during World War I. Metal detectors were further developed during World War II and have been routinely used to detect landmines since then. Due to the advancement in electronics, the sensitivity of metal detectors have improved tremendously since their beginning in World War II. A modern metal detector can detect extremely small quantities of metal under various soil types and other environmental conditions. The speed at which a handheld metal detector can be used is typically less than 1 m/s. However, the effective rate of area coverage will depend on many factors, which include the search halo of the detector, the frequency of occurrence of metal fragments and actual landmines, and the operating procedure employed.

The main and most obvious limitation of metal detectors used to detect landmines is the fact that they are metal detectors. A modern metal detector is very sensitive and can detect tiny metal fragments as small as a couple of millimeters in length and less than a gram in weight. An area to be demined is usually littered with a large number of metal fragments and other metallic debris of various sizes. This results in a high rate of false alarms since a metal detector cannot currently distinguish between the metal in a landmine and that in a harmless fragment. The more sensitive a detector is, the higher the number of false alarms it is likely to produce in a given location. Operating a detector at a lower sensitivity to reduce the number of such false alarms may render it useless for detecting

the targets it was designed to detect, that is, the small metal content of landmines buried up to a few centimeters. Electromagnetic properties of certain soils, such as magnetic soils, can limit the performance of metal detectors. This is a big problem because magnetic soils are found in most parts of the world that have landmine problems, the inability to operate satisfactorily in such soils is a significant limitation of some metal/mine detectors. Further, performance of some detectors could be severely limited by certain other environmental factors, such as accumulation of moisture on the detector head. However, this is not a limitation of the basic EMI technology. The use of EMI to detect conducting objects is also well established in other application areas such as mineral exploration, non-destructive testing, treasure hunting, security and law enforcement, and food processing.

1.2.7 Magnetic particle inspection

Magnetic particle inspection (MPI) is a NDE method used for defect detection. MPI is fast and relatively easy to apply and part surface preparation is not as critical as it is for some other NDE methods, such as LPI. These characteristics make MPI one of the most widely utilized non-destructive testing methods. MPI uses magnetic fields and small magnetic particles, such as iron filings to detect flaws in components, (Myer, 2004). The only requirement from an inspectability standpoint is that the component being inspected must be made of a ferromagnetic material such as iron, nickel, cobalt, or some of their alloys. Ferromagnetic materials are materials that can be magnetized to a level that will allow the inspection to be effective. This method is used to inspect a variety of product forms such as castings, forgings, and weldments. Many different industries use magnetic particle inspection to determine components fitness-for-use. Some examples of industries that use magnetic particle inspection are the structural steel, automotive, petrochemical, power generation, and aerospace industries. Underwater inspection is another area where magnetic particle inspection may be used to test items such as offshore structures and underwater pipelines.

MPI is based on a relatively simple concept and it can be considered as a combination of two non-destructive testing methods: magnetic flux leakage testing and visual testing. To illustrate how this method works let us consider a magnet bar. This bar has a magnetic field in and around the magnet. Any place that a magnetic line of force exits or enters the magnet is called a pole. A pole where a magnetic line of force exits the magnet is called a north pole and a pole where a line of force enters the magnet is called a south

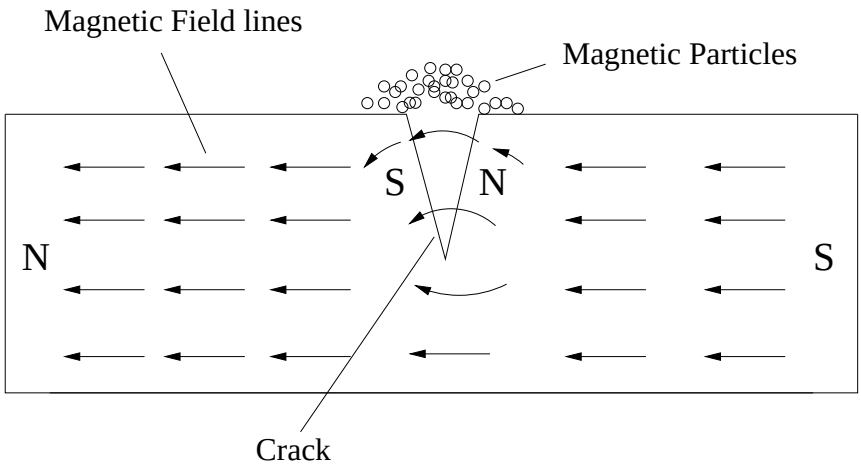


Figure 1.6: Magnetic particle inspection applied to defect detection of magnetic materials.

pole. When a bar magnet is broken in the center of its length, two complete bar magnets with magnetic poles on each end of each piece will result. If the magnet is just cracked but not broken completely in two, a north and south pole will form at each edge of the crack. The magnetic field exits the north pole and reenters at the south pole. The magnetic field spreads out when it encounters the small air gap created by the crack because the air cannot support as much magnetic field per unit volume as the magnet can. When the field spreads out, it appears to leak out of the material and, thus, it is called a flux leakage field. If iron particles are sprinkled on a cracked magnet, the particles will be attracted and cluster not only at the poles at the ends of the magnet but also at the poles at the edges of the crack, (Lovejoy, 1993). This cluster of particles is much easier to see than the actual crack and this is the basis for magnetic particle inspection, see Fig. 1.6. The first step in a magnetic particle inspection is to magnetize the component that is to be inspected. After the component has been magnetized, iron particles, either in a dry or wet suspended form, are applied to the surface of the magnetized part. If any defects on or near the surface are present, the defects will create a leakage field. The particles will be attracted and cluster at the flux leakage fields, thus forming a visible indication that the visual inspector can detect.

1.2.8 Ground penetrating radar

Ground penetrating radar (GPR) is a NDE technique that can provide high resolution images of the soil subsurface. GPR has been used during the last 20 years in civil engineering,

geology, (Knoll et al., 1991) and archeology, (Conyers and Goodman, 1997). The detection of buried landmines has been also a subject of considerable research, due to radar's potential for the detection of plastic mines which contain little or no metal, (Brunzell, 1998), (L. van Kempen and Cornelis, 2000).

GPR utilizes the transmission and reflection of high frequency, from 1 MHz to 1GHz (microwave region), electromagnetic waves within the soil. The electromagnetic wave is radiated from a transmitting antenna and travels through the material at a velocity determined by the electrical properties of the materials: dielectric permitivity, electrical conductivity and magnetic permeability. Thus, the propagation velocity of the electromagnetic waves is different in materials with different electrical properties. As the wave spreads out, and travels downward, if it hits a buried object or a boundary with different electrical properties, then part of the wave is reflected back to the surface, while part of the wave is transmitted and it continues to travel downward. The wave that is reflected back to the surface is captured by a receiver antenna. The most common display of GPR data is one showing signal versus amplitude, and it is referred to as a *trace*. A single GPR trace consists on the transmitted energy pulse followed by pulses that are received from reflecting objects. As the antennas are moved along a survey line series of traces are obtained at discrete points. All these traces form the profile of the subsurface.

A standard GPR system is composed of a transmitter and a receiver antenna, separated by a fixed distance; and a radar control unit. This pair of antennas is moved and in each movement a short pulse of electromagnetic energy is sent into the soil by the transmitter antenna. The energy reflected by the soil is recorded at the receiver antenna. The radar control unit generates the electromagnetic pulses sent by the transmitter antenna. The emitter antenna converts the electrical currents generated in the radar unit into electromagnetic waves that are propagated in the material being studied. On the other hand, the receiver antenna converts the electromagnetic waves in electrical currents that are later processed. In Fig. 1.7 such a system can be seen.

The GPR image is a representation of the interaction between the transmitted electromagnetic energy and the spatial variation in the electromagnetic properties of the soil. The link between the radar image and these electromagnetic properties is given by the Maxwell equations, that describe the propagation of electromagnetic waves. The most important issue in GPR is that the object to be detected must have different electromagnetic properties than the surrounding soil. The amount of energy reflected is especially dependent on the permitivity gap between the object and the surrounding medium, but it is also influenced

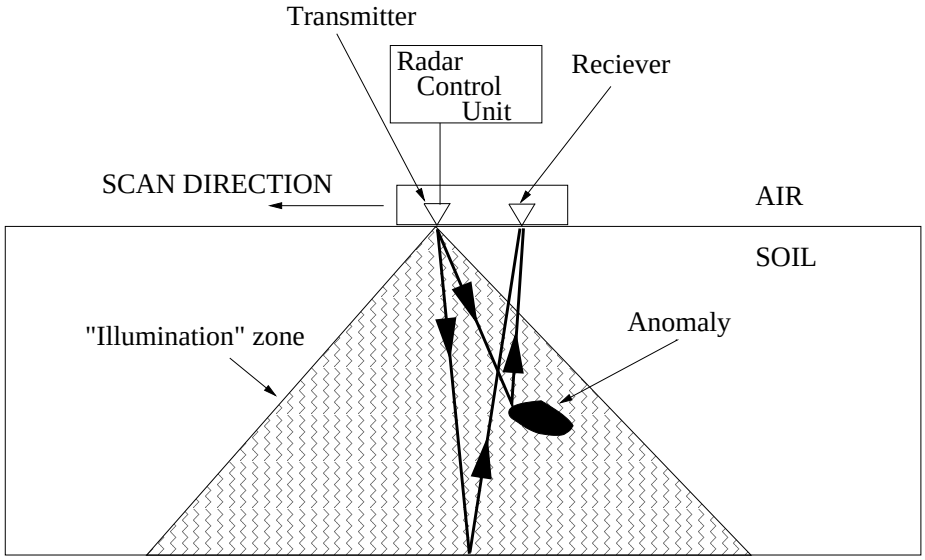


Figure 1.7: Schematic of a GPR system.

by the object's shape and size. The spatial resolution of the GPR is constrained by the frequency used. Moreover, this kind of radiation is strongly attenuated by certain types of conductive soils, such as clay, and the situation deteriorates in wet grounds where the penetration depth is very poor, (L. van Kempen and Cornelis, 2000). The main limiting factor in depth penetration of GPR is the attenuation of electromagnetic waves in the soil. The attenuation results from the conversion of electromagnetic energy to thermal energy due to the high conductivity of the soil. GPR depth of penetration can be more that 30 meters in materials having a low conductivity, an up to 5000 meters in polar ice. However, penetration is usually less than 10 meters in most soils.

Despite its great utility, the major drawback of GPR is the difficulty of detecting shallowly buried objects due to the large reflection coming from the air-soil interface, which tends to mask reflections from objects just underneath the surface.

1.2.9 Infrared thermography

Infrared thermography (IRT) technique for NDE is capable of discovering subsurface features, such as thermal properties or the presence of anomalies or defects, thanks to temperature differences observed in the surface with an IR camera, (Pramod and Inaudi, 2000). The basic premise of thermographic NDE is that the heat flow from the surface of a solid

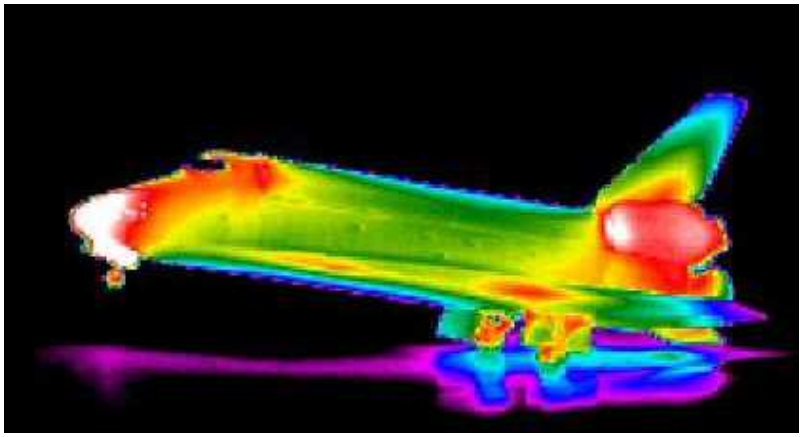


Figure 1.8: Infrared image of the space shuttle in the landing step (Image from the NASA).

is affected by internal flaws such as disbonds, voids or inclusions. This method involves the measurement or mapping of surface temperatures as heat flows to, from and/or through an object. IRT can be divided into passive and active thermography. Passive thermography test objects which are naturally at different temperatures, whereas in the active thermography an artificial stimulus is required to induce the thermal contrast.

The use of thermal imaging systems presents a series of advantages:

- Allows thermal information to be very rapidly collected over a wide area and in a non-contact mode.
- The results of the thermography are relatively easy to interpret.
- It has a wide range of applications.

However, this method also presents some difficulties:

- Difficulty to radiate uniformly a large amount of energy in short periods of time over a large surface (for active thermography).
- Cost of the IR equipment.
- Capability to measure only perturbations that result from a measurable change in thermal properties.
- It works only for near-surface thermal anomalies .

Thermal measurement methods have a wide range of uses, such as security and military applications for night vision, surveillance, navigation aid and demining; by the firemen and emergency rescue personnel for fire assessment, and for search and rescue; by the medical profession as a diagnostic tool; and by industry for energy audits, preventative maintenance, processes control and non-destructive testing, (Grande et al., 1997). This technique is also used to detect defects, (Verkhogliad et al., 2005), and to analyze corrosion damage on surfaces, (Hoffmann et al., 2001); in electronics design it is used to evaluate the dissipation of heat and to measure the temperature at the junctions, (Maldague, 1994). As an example of thermography we can see Fig. 1.8 where an IR image of the space shuttle in the landing step is showed; from this image a study of the effects of the temperature in the landing can be done.

1.3 NDE techniques for buried landmines detection

According to official reports, more than 40 million landmines are buried around the world. Although land mine problems existed in many regions, the deployment of U.S. troops to Bosnia in 1995 and to Afghanistan in 2001 gave the land mine issue a particular sense of urgency. Intended for warfare, these mines remain buried after the end of the conflict. These mines are triggered by civilians causing around 15,000-20,000 victims per year in 90 countries, (ICBL, 2006). The U.S. State Department estimates that there are around 40-50 million of buried mines that need to be cleared. According to (Horowitz, 1996) 100,000 mines are found and destroyed per year; thus 450 years will be necessary to clean all mines. However, each year, 1.9 million of new mines are buried. Thus it is necessary to develop new techniques which allow to detect mines quickly and with high precision. The presence of mines also causes economic decline being one of the major limitations to agricultural work on these regions, (Cameron and Lawson, 1998). In 1997 was signed the *Ottawa treaty*, (Ott, 1997), to ban the production and use of AP mines; in 2007 this treaty was signed by 158 countries, (ICBL, 2008), however the most important AP manufacturers, China, Russia, India and EE.UU, have not yet signed it.

Nowadays more than 350 types of mines exist, (Vines and Thompson, 1999); however they can be divided into two main categories:

- Antipersonnel (AP) mines.
- Antitank (AT) mines.

AT mines are relative big and heavy (2-5 Kg) and are usually laid on the ground forming a regular patterns shallowly buried. AT mines have enough explosive to destroy a tank or a truck, as well as to kill people in or around the vehicle; they also require more pressure to be detonated than AP mines. On the contrary AP mines contain less explosive and are lighter than AT mines. AP mines can be buried anywhere, they may lie on the surface or be shallowly buried. Sometimes they are placed in a regular pattern to protect AT mines, however in most cases they are placed random. Moreover, as AP mines are light and small, wind or rain can easily move them making their location, even with the original pattern, more difficult. AP mines are designed to damage foot soldiers avoiding their penetration into an specific area. These mines can kill or disable their victims and are activated by pressure, tripwire or remote detonation. These characteristics make AT mine detection and clearance easier than AP mine detection.

Detection and clearance of buried mines is a big problem with lots of humanitarian, environmental and economic implications. Current techniques for non-destructive evaluation of soils fail to address the detection of small plastic landmines, which are the most difficult to detect. There is no universal technique capable of detecting buried landmines in all situations. The most widespread techniques in mine detection are metal detector and magnetometers. Magnetometers are used to detect ferromagnetic objects and they measure the disturbance of the earth's natural electromagnetic field. Many modern mines, see Fig. 1.9, have almost no metal parts except for the small striker pin. Although metal detectors can be tuned to be sensitive enough to detect these small items (current detectors can track a tenth of a gram of metal at a depth of 10 cm), such sensitivity detects more metal debris and increases considerably the rate of false alarms. Increasing the sensitivity of metal detectors, therefore, does not solve the problem of non-metal mines satisfactorily. Taking in mind this limitation new techniques are appearing to detect the plastic landmines. Several techniques have been proposed for mine detection, such as acoustic techniques, (Sabatier and Xiang, 2001); X-rays techniques, (Lockwood et al., 1997); biosensors, (Larsson and Abrahamsson, 1993); ion mobility spectrometers, (Jankowski et al., 1992); nuclear quadrupole resonance, (Englelbeen, 1998); neutron analysis, (Bach et al., 1996); GPR and IRT. Each technique has its advantages and disadvantages, for a more detailed description of theses techniques see (Gros and Bruschini, 1998; Siegel, 2002; López, 2003).

In the following section we will make a comparison between two of the most promising techniques to detect plastic landmines: GPR and IR thermography.

Figure 1.9: A typical low metal antipersonnel mine (Type 72).

1.3.1 IRT versus GPR

IR thermography, (Khanafer and Vafai, 2002; López et al., 2004; Thanh et al., 2006, 2007) and GPR, (Brunzell, 1998; Kosmas et al., 2002; Siegel, 2002; Savelyev et al., 2007) are two of the most promising techniques for landmine detection. GPR is based on the emission of electromagnetic waves in the radar domain into the soil and study soil's response; whereas IR thermography is based on the study of the thermal evolution of the soil, where IR cameras are used to obtain such an information. The presence of a mine in the soil produces a disturbance in the soil response to both radar and thermal stimulus. It is obvious that the mine detection can only be performed if the mine and the soil have different electrical properties (GPR) or thermal properties (IRT).

GPR has been used during the last 20 years in civil engineering, geology, and archeology to detect buried objects and to analyze soil and therefore this technology is well-researched. In GPR electromagnetic waves are emitted into the ground, through an emitter antenna. Reflections from the soil caused by dielectric variations (such as the presence of an object) are measured. By moving the antenna an image that represents a vertical slice of the soil can be reconstructed; further data processing allows the display of horizontal slices or three-dimensional representations, (Daniels, 1996). This technology has limitations; in particular, the resolution needed to detect small objects involves GHz frequencies, which decreases soil penetration and increases image clutter. Another constraint is cost, the price of current equipment is a limitation for humanitarian applications, compared to the cost of standard equipment. In order to decrease the size and price of this type of sensor the Lawrence Livermore National Laboratory (LLNL) has developed and patented the

Micropower Impulse Radar (MIR). The small footprint of the antennas (less than 50 cm^2) should allow to build quite easily an array for a faster and simplified scan of the minefield, (Azevedo et al., 1995). A vehicle with GPR system is described in (Siegel, 2002). This is an autonomous vehicle with a Global Positioning System (GPS) which gives the position of the mines founded during the exploration of the soil. Another limitation of GPR is that the reflection that occurs at the soil-air interface tends to mask the reflections that come from objects just beneath the surface and this makes the detection of shallowly buried landmines difficult.

The use of IRT for landmine detection consists of subjecting the soil to a thermal stimulus (natural or artificial) and studying its thermal evolution. Mines retain or release heat at a different rate from their surroundings and an infrared camera is used to acquire images that reveal the thermal contrast that appears on the surface when there is a buried object underneath. The application of IR imaging to mine detection presents some problems; for example, IR imaging requires sensitive cameras ($\Delta T < 0.1$ ℃) with sufficient spatial resolution; this technology manly allows to detect mines at a maximum burial depth of 10-15 cm, at higher depths the perturbation caused by the mine is not reflected at the surface. In addition, the results of passive infrared images depend heavily on environmental conditions, (Russell et al., 1997), and during cross-over periods (morning and evening), the thermal contrast is negligible and thus mines undetectable through IR in these periods. This method is quick and can cover a wide area at a time.

In IRT it is not essential to provide a pulse of heat since there will always be natural thermal gradients in the soil caused by the daily changes in temperature as the sun rises and falls; this is in opposition to GPR where it is necessary to generate an artificial stimulus. The water content of the soil and the presence of vegetation are important factors in the temperature differences observed between soil and mine, but these factors affect both techniques.

Summarizing, GPR techniques constitutes one of the most promising alternatives in mine detection, with excellent detection results for deeply buried mines, but shows a severely degraded performance at shallow depths due to reflections coming from the air-ground interface, (L. van Kempen and Cornelis, 2000). The maximum range of applicability of the IRT is around 10-15 cm, where it has been proved as an efficient technique. Thus, some mine detection techniques use sensor fusion to obtain better results in the detection, (Cremer et al., 2001; Beaven et al., 2004). We have chosen IRT as landmine detection technique due to the fact that this kind of mines is usually buried at low depths

to be activated when a person is near the mine.

1.4 Infrared thermography for landmine detection

This is an attractive technique for some mine detection tasks because it can be used from a considerable standoff distance, it provides information on several mine properties, and it can rapidly survey large areas. Its ability to detect mines has been recognized since the 1950s, (Maksymonko and Le, 1999). IRT sensors respond to electromagnetic radiation in a sensor-specific wavelength range. The source of the received signal may be either natural (i.e., thermal emission from the target or scattering of sunlight) or artificial (e.g., an infrared illuminator), which leads to both passive and active sensor concepts. We will focus on the passive thermography which is based on the heating effect of the solar radiation and the cooling process that the soil undergoes during the night. Each material shows a characteristic thermal response to a given stimulus, also known as the *thermal signature*. Thus, the cooling or heating process affects to buried objects and the surrounding soil in a different way. This difference is due to the fact that the mines is a better insulator than the soil. Thus, during the day the soil layer over the mine tends to accumulate thermal energy because the mine blocks the transport of thermal energy. As a result of this process the soil over the mine tends to be warmer than the surrounding soil. In the evening, the soil over the mine gives up thermal energy faster than the surrounding soil and results cooler. Around midday the soil and the mine reach the thermal equilibrium, which makes impossible to perform the detection in this temporal gap. The main limitation of this technique is the fact that temperature differences strongly depends on the atmospheric conditions. On the other hand, active thermography uses artificial energy to heat the soil under study, avoiding the dependence on the atmospheric conditions.

In spite of their long history, there is little compelling performance data available for infrared detection of antipersonnel mines. Two likely reasons for this are as follows: first, infrared mine signatures tend to be highly dependent on environmental conditions, which complicates data collection and performance assessment. Indeed, a number of multisensor mine detection systems have included an electro-optical (EO) sensor, which includes both infrared bands and hyperspectral sensors, but have not used it because of unreliable data quality. As a result, much of the work to date comprises concept demonstrations and studies of specific phenomena. Second, the only EO performance tests involving statistically meaningful sample sizes have been conducted on U.S. Army test sites containing primarily

antitank mines. Detection of antipersonnel mines is significantly more challenging than detection of antitank mines.

The use of IRT as NDE for landmine detection consists of subjecting the area under inspection to a source of natural or artificial heating/cooling process and studying the soil's response by means of the analysis of its thermal evolution given by a temporal sequence of infrared images. In this sense the study of the basic phenomenology lead to the development of mathematical models of the soil, (Kahle, 1977; England, 1990; England et al., 1992; Liou and England, 1998; Pregowski et al., 2000). The idea underneath is to characterize, and therefore predict, the thermal behavior of the unperturbed soil under given conditions, i.e., its *thermal signature*. The presence of buried mines or other objects will alter this signature, which if manifested as a thermal contrast on the surface. From the analysis of this thermal contrast and, in particular of its dynamic evolution, it is possible to extract relevant information about the nature of the objects. A deep analysis of the evolution of the thermal contrast over a diurnal cycle can be found in (Khanafer and Vafai, 2002), where the conditions of for detection are studied, and sunrise and sunset are established as the periods of the day in which the presence of buried mines induces a greater thermal contrast on the surface (around 4-6℃).

An efficient way of extracting information from this data regarding the presence of landmines using a 3D thermal model of the soil based on the solution of the heat equation was presented in (López, 2003; López et al., 2004, 2008). The process is divided in two steps. On the first one, the forward problem, the soil is subjected to a heating process and a comparison between temperatures measured at the soil surface (through IR imaging) and those obtained by simulation using the model under the assumption of homogeneous soil and mine absence is made. The differences between measured and simulated data put into evidence the presence of unexpected objects on the soil. The second step is an inverse engineering problem where the thermal model must be run for multiple soil configurations representing different types of possible targets (mine, stone, ...) and depths of burial. The nearest configuration to the measured data give us the estimated nature and location of the targets. This approach and, particularly, the inverse engineering process, makes an intensive use of the 3D thermal model that needs to be solved iteratively involving complex, coupled sets of partial differential equations.

In this section we will introduce the mine detection using IRT following the methodology proposed in (López, 2003; López et al., 2004, 2008). First, we will show the thermal processes considered and the mathematical formulation of the thermal model used to per-

form the detection. After this, a description of the inverse problem used to detect the nature and position of unexpected object buried in the soil is made.

1.4.1 Physical phenomena

In this section we will describe the physical phenomena that have been considered in the thermal model:

- Nature of the infrared radiation.
- Convection process between the soil and the air.
- Incident sun radiation in the soil surface.
- Infrared radiative heat exchange between the atmosphere and the soil.
- Radiative exchange processes.

All these phenomena occur at the soil-air interface and affect the internal piece of soil through a conduction phenomena, that drives the heat transfer inside the soil volume.

Infrared Radiation

The electromagnetic spectrum is a continuum of all electromagnetic waves arranged according to frequency or wavelength, see Fig 1.10. As we can see the infrared region goes from 1 μm to 100 μm. The infrared sensors are based on the fact that all objects whose temperature is above absolute zero (-273.15 ℃) emit radiation and that bodies at ambient temperature emit the largest amount of radiation in the infrared region. The radiation emitted by an object at a given wavelength depends of its temperature and it is given by the *Planck's law of black body radiation*:

$$E_{\lambda,T} = \frac{C_1}{\lambda^5} \frac{1}{e^{\frac{C_2}{\lambda T}} - 1} \tag{1.1}$$

where $E_{\lambda,T}$ is the black body spectral emissive power; $C_1 = 3.742 \cdot 10^{-16}$ Wm^2 and $C_2 = 1.439 \cdot 10^{-2}$ m/K; λ is the wavelength and T is the temperature. The spectral variation of the emissive power with the wavelength for various temperature values can be seen in

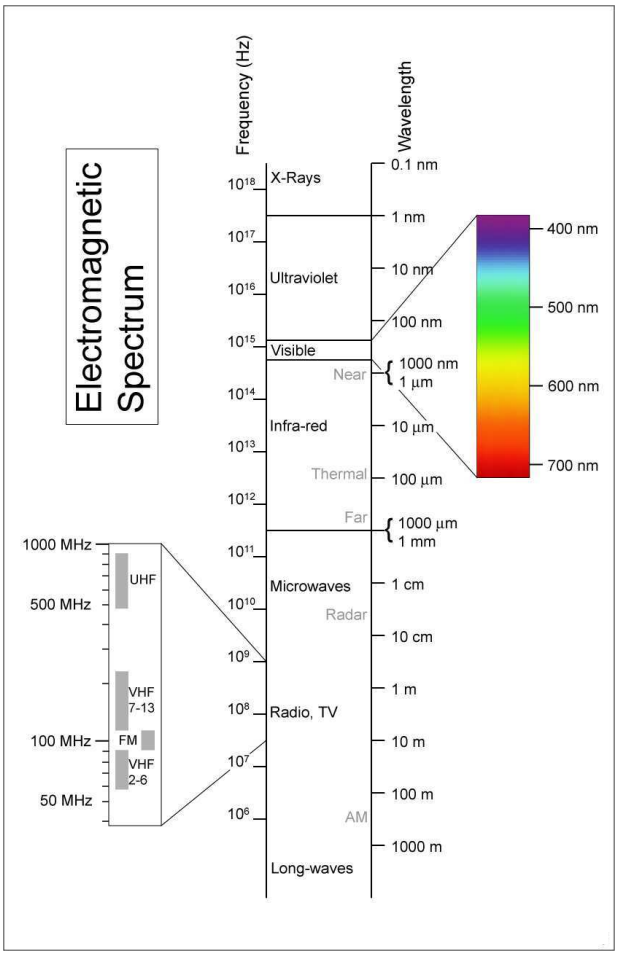


Figure 1.10: The electromagnetic spectrum.

Fig. 1.11. The emissive power increases dramatically with the temperature and a maximum of radiation is emitted for a characteristic wavelength at a given temperature:

$$\lambda_{max} T = 2.898 \cdot 10^{-3} \quad \text{mK} \tag{1.2}$$

this equation is known as *Wien's displacement law* and its meaning is shown in Fig. 1.11. As the temperature of the object increases, the wavelength of the maximum emission decreases. If Eq. 1.1 is integrated over all frequencies, the Stephan-Boltzman law is obtained:

$$E_T = \sigma T^4 \tag{1.3}$$

where E_T is the total emissive power and $\sigma = 5.67 \cdot 10^{-8}$ W/m^{2}K^4 is the Stephan-Boltzmann constant. However, real objects never behave as ideal black bodies and the emitted radiation at a given frequency is a fraction of what the emission at the same temperature would be for black body; this fraction is called the emissivity. The emissivity of a material

specifies how well a real body radiates energy as compared with a black body. This emissivity depends on factors such as temperature, emission angle, and wavelength and can be defined as:

$$\epsilon(\lambda, T) = \frac{E_{\lambda,object}(T)}{E_{\lambda,BB}(T)} \tag{1.4}$$

where $\epsilon(\lambda, T)$ is the emissivity, $E_{\lambda,object}$ is the amount of energy emitted by the object at a given temperature and $E_{\lambda,BB}$ is the amount of energy emitted by the black body at the same temperature. In many cases the emissivity is a complex function of the wavelength; however, sometimes for certain wavelength ranges the following approximation can be done:

$$\epsilon(\lambda, T) \simeq \epsilon(T) \tag{1.5}$$

This is known as the gray body approximation and it is valid in regions where the dominant values of the emissivity are located in the wavelength region that is responsible for the bulk of the blackbody emissive power σT^4. The infrared radiation corresponds wavelengths located between $1\mu m$ and $100\mu m$, as can be seen in Fig. 1.10, and according to Planck's law all objects at ambient temperature ($\sim 300\,K$) emit their radiation peak in the infrared region, see Fig. 1.11. Thus in this case we can use the gray body approximation and a constant value of the emissivity can be used.

Thermographic cameras detect radiation in the infrared range of the electromagnetic spectrum and produce images of that radiation. The amount of radiation emitted by an object increases with temperature, therefore thermography allows to see variations in temperature. With a thermographic camera warm objects stand out well against cooler backgrounds. The two parameters of performance for any thermographic system are *sensitivity* and *resolution*. Sensitivity is the capacity of a sensor to discriminate small radiance changes in an object. For infrared cameras, this is usually described in terms of an equivalent blackbody temperature. Ideally, an infrared sensor would be able to resolve radiant changes in targets and backgrounds to an infinite number of equivalent blackbodies temperatures. However, system noise and sensor dynamic range eliminate any chance of infinite sensitivity. On the other hand, resolution refers to how well a sensor can detect small spatial details in the object. Sensitivity and resolution are related and normally conflicting. A combined sensitivity and resolution performance parameter was developed, (Driggers et al., 1999), called the minimum resolvable temperature difference (MRTD or just MRT) that accurately describes the interrelationship between sensitivity and resolution for a particular sensor. MRT combines both the sensor and the operator characteristics into one measurement to define the sensors ability to display a structured target with small temperature differences. The MRT is defined, (D'Agostino and Scott, 1993; Ratches et al.,

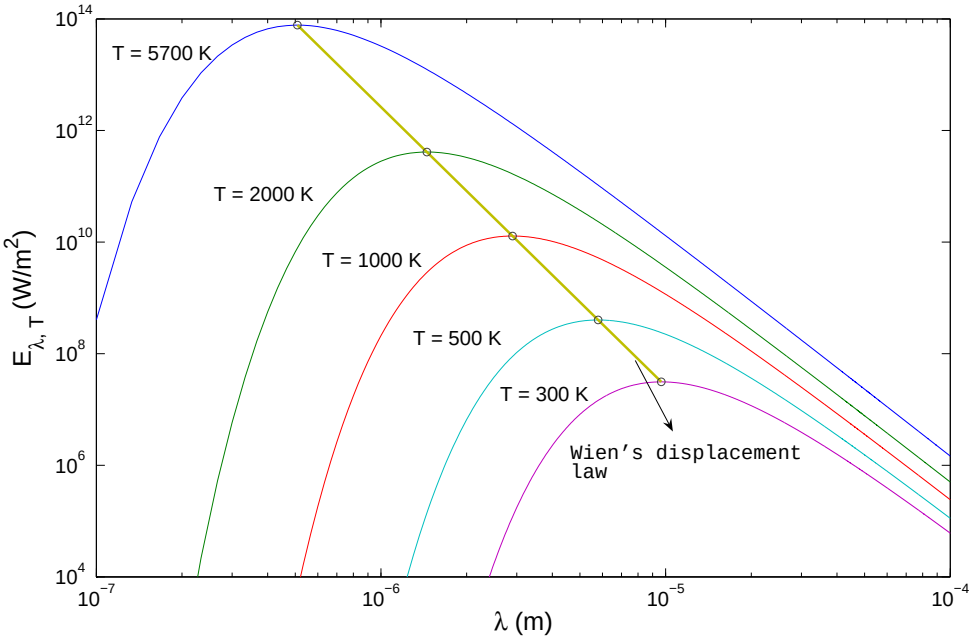


Figure 1.11: The effects of wavelength and temperature on the black body emissive power and Wien's displacement law.

2001), as the differential temperature of a four-bar target that makes the target just resolvable by a particular sensor. It provides sensor sensitivity as a function of the four-bar target frequency (i.e., resolution) and it is the primary measurement of performance for infrared imaging systems. The MRT can be calculated analytically, (Ratches et al., 2001), however, due to its probabilistic nature and also considering the ability of human operators, it is often determined experimentally by adjusting various spatial frequency bar patterns and noting the ability of a trained operator to resolve them.

The temperature resolution of commercial infrared cameras is usually about 0.03-0.06 ℃. Other parameters of thermographic systems are the maximum and minimum temperatures that can be measured; usually, this range goes from -40 ℃ to 500 ℃.

Convection

Convection is a heat transfer mechanism that occurs when a fluid is in contact with a solid surface. The fluid acts as a carrier for the energy that it draws from (or delivers to) a solid surface. Convection is a heat transfer mechanism in which the characteristics of the flow,

such as turbulence or velocity distribution, affect greatly the heat transference between the surface and the fluid. The convection process involves two mechanisms: energy transfer due to random molecular motion (diffusion) and macroscopic motion of the fluid, in our case air. The convection term can be expressed as, (Kahle, 1977):

$$q_{conv}(t) = \rho_a c_{pa} C_d (W(t) + 2)(T_{air}(t) - T_{soil}(t)) \tag{1.6}$$

where ρ_a [Kg/m^3] is the air density, c_{pa} [J/K] is the specific heat of the air, C_d is the wind drag coefficient (chosen to be 0.002), and $W(t)$ [m/s] is the wind speed. In most practical cases this expression can be transformed into, (Incropera and DeWitt, 2002; Khanafer and Vafai, 2002):

$$q_{conv}(t) = h(T_{air}(t) - T_{soil}(t)) \tag{1.7}$$

where h [W/m^2 K] is the convective heat transfer coefficient that is known to be strongly dependent on the wind speed. This expression is known as *Newton's law of cooling*. As we will shown, the landmine detection procedure makes use of data restricted to a short time range and it is therefore possible to assume a constant value of h based on an average wind speed ($h = 5$ W/m^2 K based on an average wind speed of 2 m/s).

Solar Radiation

Energy comes from the Sun in the form of radiation, in a range of wavelengths whose maximum is in the visible part of the spectrum. It is the light from the Sun, reflected off objects, that we are able to see, and the ultraviolet radiation from the Sun will damage our skin (for wavelengths less than about 0.295 nm). These forms of radiation have no other significant sources on the Earth, and consequently are negligible at night.

The intensity of the incoming radiation from the Sun is fairly constant at the top of the atmosphere (there is some variation over the year due to the Earth's orbit). The spectral distribution of solar radiation at the top of the atmosphere is approximately equal to a blackbody at 5900 K, see Fig. 1.12, and it is concentrated in the short wavelength region ($0.2\mu \leq \lambda \leq 3\mu$m), with the peak at approximately 0.50 μm. It is this short wavelength concentration that excludes the assumption of gray body behavior for solar irradiated surfaces, because emission is generally in the spectral region beyond 4 μm and it is not true for almost all substances that spectral properties are constant over such a wide spectral range. In Fig. 1.13 the variation of sand reflectivity, that it is equal to its

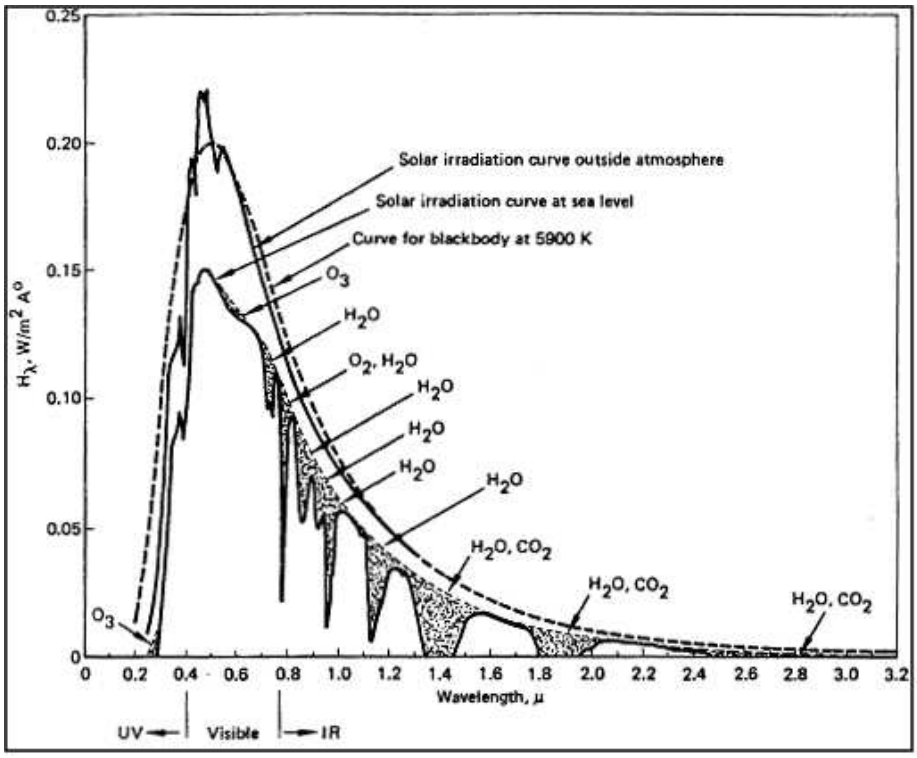


Figure 1.12: Spectral distribution of solar radiation (reprinted from (Wolfe and Zissis, 1978)).

emissivity (Incropera and DeWitt, 2002), is shown and it can be noted that there is a big variation between the emissivity of the sand in the short-wave region (until $2\,\mu m$) and in the long-wave region ($10 - 20\,\mu m$). As solar radiation passes through earth's atmosphere, its magnitude and spectral composition change significantly. This change is mainly due to absorption and scattering of the radiation by atmospheric components (H_2O, O_3, CO_2), see Fig. 1.12. The short-wave radiation emitted by the sun can be estimated using this expression:

$$q_{sun} = (1 - al)S_0(1 - C)M(Z)H(t) \tag{1.8}$$

where $S_0 = 1353$ W/m^2,the solar constant, is the flux of solar energy incident on a surface normal to the sun's rays when the earth is at its mean distance from the sun; al is the ground albedo, C is a factor that takes into account the reduction in solar flux due to the cloud cover and $H(t)$ is the local insolation function, (Sendur and Baertlein, 2000). At the top of the atmosphere, this insolation depends on the latitude and the time of day and can be expressed as a time function as:

$$H(t) = \left(\frac{\overline{R}_{S-E}}{R_{S-E}}\right)^2 cos(\theta(t)) \tag{1.9}$$

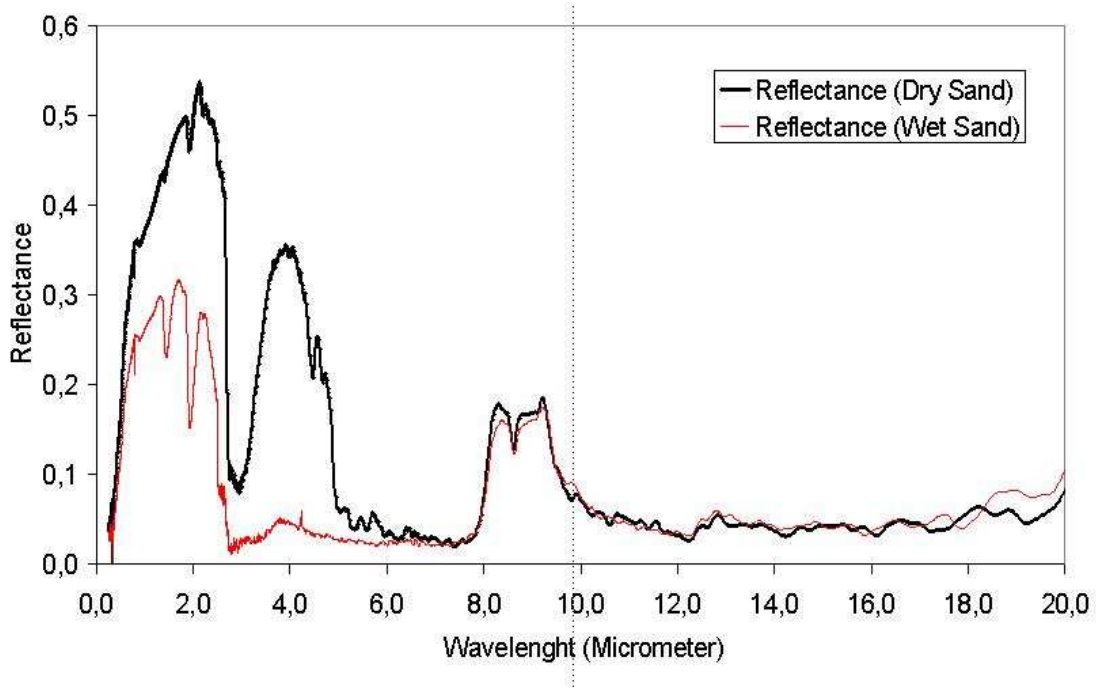


Figure 1.13: Variation of sand reflectivity with the wavelength (courtesy of the ETRO-IRIS Department from Vrije Universiteit in Brussel).

where $\overline{R}_{S-E}$ is the mean sun-earth distance; R_{S-E} is the sun-earth distance at the time of observation and $\theta(t)$ is the solar zenith angle. This function depends on the eccentricity of the earth orbit; at the perihelion of the orbit $\frac{\overline{R}_{S-E}}{R_{S-E}} = 0.983$, meanwhile that at the aphelion $\frac{\overline{R}_{S-E}}{R_{S-E}} = 1.017$. Thus, the variation of the solar insolation function at the top of the atmosphere is 93 W/m^2. The other principal factor that defines solar insolation received by a horizontal surface at the top of the atmosphere is the solar zenith angle θ:

$$cos\theta = sin\phi\, sin\delta + cos\phi\, cos\delta\, cosh \tag{1.10}$$

where ϕ is the latitude, δ is the declination and h is the hour angle. The declination, δ, can be approximated using the following approximation:

$$\delta \simeq -23^o 27' \, cos\left(\frac{360}{360.25}\,(JD + 9)\right) \tag{1.11}$$

where JD is the Julian Day. The hour angle, h, is defined by $\pm 15^o$ each hour before or after solar noon. However q_{sun} can be directly measured with low cost equipment reducing the complexity of the problem and the the expression of q_{sun} can be reduced to:

$$q_{sun} = \epsilon_{sun} q^e_{sun} \tag{1.12}$$

where ϵ_{sun} is the absortivity of the soil for the short-wavelength solar flux and q^e_{sun} is the experimental measured solar radiation.

Radiative processes

Due to the fact that the soil and the air have temperatures higher that -273℃ they emit radiation. Thus, there is a radiation exchange processes between the soil and the the air due to their different temperatures. The term q_{rad}, that takes into account this process, can be expressed as:

$$q_{rad}(t) = q_{sky}(t) - q_{soil}(t) \tag{1.13}$$

where $q_{sky}(t)$ is the long-wave radiation from the atmosphere given by the Stephahn-Boltzmann law, $q_{sky}(t) = \sigma\epsilon T^4_{sky}$, where T_{sky} is the effective sky radiance temperature, that can be estimated as, (England, 1990):

$$T_{sky}(t) = T_{air}(t)(0.61 + 0.05\sqrt{w})^{0.25} \tag{1.14}$$

where $T_{air}(t)$ is the air temperature and w is the water vapor pressure in mmHg, (Kahle, 1977). If moisture transfer phenomena are not considered, then Eq. (1.14) can be simplified to:

$$T_{sky}(t) = 0.9\ T_{air}(t) \tag{1.15}$$

The air temperature can be also estimated using this expression, (England, 1990):

$$T_{air}(t) = T_{0,air} - T_{del}\ cos\left(\frac{2\pi(t-2)}{24}\right) \tag{1.16}$$

where $T_{0,air}$ and T_{del} are values estimated from local meteorological data and t it the local time (in hours) measured from midnight.

The heat loss of the soil due to the radiative process can be calculated as $q_{soil} = \sigma\epsilon T^4_{soil}$ where ϵ is the emissivity of the soil and T_{soil} is the temperature of the soil at the soil-air interface. This is a non-linear term but it can be linearized for long-wavelength as, (Watson, 1973):

$$q_{rad}(t) = \sigma\epsilon T^4_{sky} - \sigma\epsilon T^4_{soil} \simeq \sigma\epsilon T^3_{sky}(T_{sky} - T_{soil}) \quad for\ \frac{T_{sky} - T_{soil}}{T_{sky}} << 1 \tag{1.17}$$

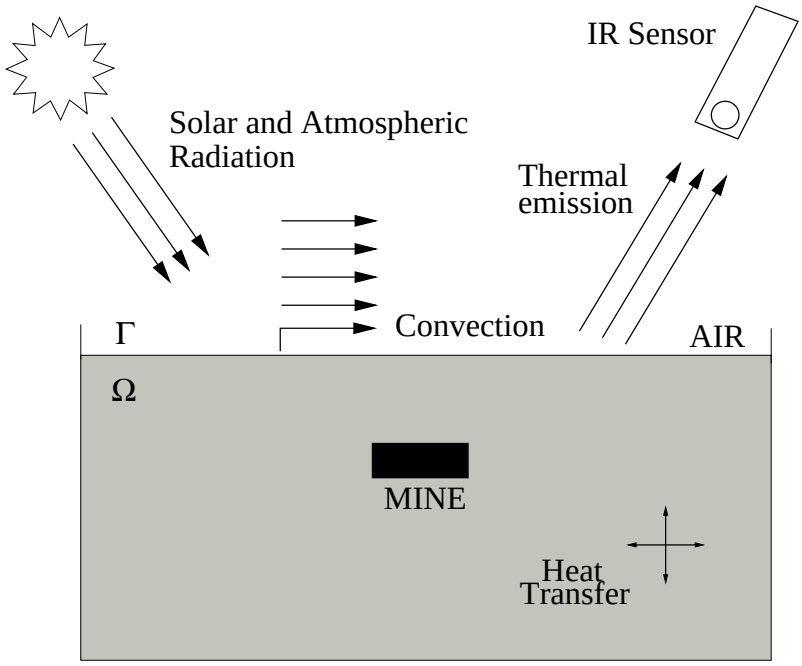


Figure 1.14: Different heat transfer processes taking place in the soil volume and at the air-soil interface.

1.4.2 Thermal Model

The processes considered in the thermal model have been described in the previous section and can be seen in Fig. 1.14, where Ω is the considered soil volume and Γ is the soil-air interface. The thermal model used in this work is a 3-D model and both the soil and the mines are considered as isotropic objects, which simplifies a lot the model and its mathematical formulation. We also assume that the temporal variation of the moisture content and the mass transference during the time of analysis are negligible. As the scope of applicability of IRT techniques for soil inspection is limited to a maximum depth of roughly 10-15 cm and given that our analysis is limited to a short period of time around sunrise or sunset, the thermal parameters can be supposed to be constant during the interval of interest. The equation that describes the thermal behavior of an object is called the Conduction Equation or Heat Equation. Under these assumption the overall process is described by the time-dependent single-phase 3D heat equation:

$$\frac{\partial T(\overrightarrow{r},t)}{\partial t} - div\,(\alpha(\overrightarrow{r})\; grad\, T(\overrightarrow{r},t)) = 0 \;\text{ with } \alpha = \frac{k}{\rho c_p} \tag{1.18}$$

where $\overrightarrow{r} = (x, y, z)$, ρ [kg/m^3] is the density, c_p[J/kg K] is the specific heat, k[W/m K] is the thermal conductivity, α [m^2/s] is the thermal diffusivity and T [K] is the distribution of temperatures inside the soil. To solve this equation the initial and boundary conditions are required:

$$k\frac{\partial T(\overrightarrow{r},t)}{\partial n} = q_{net}(t) \quad for\ \tau\ \times\ \Gamma \tag{1.19}$$

$$\frac{\partial T(\overrightarrow{r},t)}{\partial n} = 0 \quad for\ \tau\ \times\ \partial\Omega\backslash\Gamma \tag{1.20}$$

$$T(x, y, z \to \infty, t) = T_\infty \tag{1.21}$$

$$T(\overrightarrow{r}, t = t_0) = T_0(\mathbf{r}) \quad in\ \ \Omega \tag{1.22}$$

where $\tau = t_0, ..., t_f$ is the time interval of analysis, q_{net} is the net heat that flows through the soil-air interface Γ, n is the normal to the surface under consideration $\partial\Omega$. Eq. (1.19) gives the boundary condition in the air-soil interface; Eq. (1.20) shows the boundary conditions applied to the other sides of the volume under consideration, imposing a vanishing heat flux across them, which is valid for a big enough soil volume. Eq. (1.21) is the deep-ground condition and it establishes that the temperature remains constant for a large enough depth, this means that the phenomena that are taking place at the soil-air interface do not affect the temperature of the soil from this depth. Finally, Eq. (1.22) gives the initial conditions for the system. As can be seen in Fig. 1.14, convective and radiative process drive the thermal behavior of the soil at the soil-air interface. These phenomena are introduced in the thermal model via the boundary conditions, see Eqs. (1.19)-(1.21). As we consider convective and radiative process, the net heat flux at the soil-air interface can be written as:

$$q_{net}(t) = q_{sun}(t) + q_{rad}(t) + q_{conv}(t) \tag{1.23}$$

where q_{net} is the net heat flux in the direction of the normal to the soil surface accessible for measurements; q_{sun} is the short-wave radiation emitted by the sun and absorbed by the soil; q_{conv} is the heat flux exchange due to a convection process between the air and the soil at the soil-air interface; and q_{rad} is the heat flux exchange due to radiative process.

As was mentioned in previous sections, the first term in Eq. (1.23), the solar flux, can be estimated in a straight-forward manner knowing the ground albedo and the local sun irradiation function. However, in experimental setups it can be easily measured with the help of low-cost equipment which significantly reduces the complexity of the problem and, therefore, the expression of q_{sun} can be reduced to Eq. (1.12). The second term in Eq. (1.23), that accounts for the heat transfer due to radiative processes, can be linearized using the approach given by Eq. (1.17). The last term in Eq. (1.23) concerns the convection

process that takes place at the soil-air interface. Thus, all terms of the boundary conditions can be easily obtained which a low-cost equipment to measure the air temperature and the sun irradiation, avoiding the necessity of been calculated. In conclusion, the heat transfer terms due to radiation and convection are unequivocally determined when the temperature of the air and the soil are known. These quantities can be easily measured, which avoids inaccuracies from an estimation of the atmospherics parameters. The heat equation can be solved numerically using the Finite-Difference Time-Domain (FD-TD) method , (Bejan, 1993; Incropera and DeWitt, 2002), that is the most widely used method to obtain the numerical solution of partial differential equations (PDE) equations and it will be studied in detail in Chapter 2.

1.4.3 Non-destructive soil inspection

Many physical systems are characterized by the solution of a differential equation or system of equations subject to known boundary conditions. This is the called *forward problem*. The representation of a non-linear forward operator can take the form of a functional equation involving a map F, which represents the connection between the model and the data:

$$y = F[p] \tag{1.24}$$

where F is a non-linear operator between Hilbert space Y and P, y is the measured data and p the original distribution of parameters that gives rise to y under the application of operator F. An *inverse problem* of this will be the reconstruction of the original distribution of parameters based on the measurements of the resulting data.

Solving an inverse problem implies approximating the best solution $p^{\dagger} = F^{-1}[y]$ of Eq. (1.24). In general, $y \in Y$ is never known exactly but up to an error of $\delta \neq 0$. Therefore, we assume that we know $\delta > 0$ and $y^{\delta} \in Y$ with $\|y - y^{\delta}\| \leq \delta$. Thus, y^{δ} is the *noisy* data and δ is the noise level. Under this situation is, generally, impossible to compute numerically a solution of the problem unless making use of the regularization techniques, which makes it possible to restore stability and existence/uniqueness of the solution and develop efficient numerical algorithms, (Kirsch, 1996).

The non-linear Landweber iteration method is defined by the following recursive procedure, (Engl et al., 1996):

$$p_k^{\delta} = p_{k-1}^{\delta} + F'[p_{k-1}^{\delta}]^T (y^{\delta} - F[p_{k-1}^{\delta}]) \quad k \in \mathbb{N} \tag{1.25}$$

As initial guess $p_{k=0} = p_0$ is set, incorporating a *priori* knowledge of an exact solution $p^\dagger$. For non-linear problems, additional conditions about the stopping rule have to be imposed to guarantee convergence rates, (Engl et al., 1996). The inequality known as the *discrepancy principle* will be used to define the stopping index $k(\delta, y^\delta)$ with

$$\|y^\delta - F[p_{k(\delta,y^\delta)}]\| \leq \mu\delta < \|y^\delta - F[p_k^\delta]\| \quad 0 \leq k \leq k(\delta, y^\delta) \tag{1.26}$$

if the parameter μ is chosen subject to the constraint

$$\mu > 2\frac{1+\eta}{1-2\eta} \tag{1.27}$$

being $\eta < 1/2$. The proof that this stopping rule regularizes the Landweber iteration method can be found in (Engl et al., 1996).

Once the convergence criteria and the stopping rule have been established, the iterative procedure is applied to update the estimate of the solution in practical situations. It can be seen that the update term in the Landweber iterative method, see Eq. (1.25), is the negative gradient at $p = p_{k-1}^\delta$ of the functional

$$J = \|F[p] - y^\delta\| \tag{1.28}$$

that is the misfit between the model and the measured data. Solving the inverse problem is hence equivalent to minimize this functional.

The solution of the forward problem, F, is then the solution of the system of Eqs. (1.18)-(1.21) given the boundary conditions in the surface of the soil, Γ, and a distribution of soil parameters p within the volume Ω. According to Eq. (1.18), p corresponds to the value of the thermal diffusivity at every point within the soil volume, $\alpha = \alpha(x, y, z)$. Given the 3D nature of the model, the application of the operator F to the set of parameters, p, produces a 3D solution where the distribution of temperatures both on the surface and within the soil volume is calculated. For the sake of clarity we denote as $F[p]$ the distribution of the temperatures on the surface of the soil, which is compared with the temperature distribution acquired by the IR camera, our noisy data y^d. The reconstruction of the internal composition of the soil can be viewed as an inverse problem of this,that can be formulated as: *given the boundary conditions of the system and a distribution of measurements y on Γ, derive the soil parameter distribution, p, within Ω, that is, $p = F^{-1}[y]$.*

Now we will introduce the procedure for the non-destructive inspection of soils for the identification and classification of mines and mine-like targets on infrared images based

on the solution of the heat equation and the use of inverse problems techniques, (López, 2003; López et al., 2004, 2008). The process starts with the acquisition of a sequence of infrared images of the surface of the soil under known heating and atmospheric conditions. As explained before, sunrise and sunset are the preferred times for detection. We will also assume that a pre-processing stage is run on a conventional PC in order to align the images and map grayscale colors to temperature values on the surface. Next, the soil inspection procedure itself starts. First, we run a detection procedure, as will be explained in the following section, to obtain the mask of potential targets. Then, a quasi inverse process operator is used to identify the presence of antipersonnel mines among the potential targets. For those targets that failed to be classified as mines (and are therefore labelled as *unknown*), a full inverse procedure to extract their thermal diffusivity will be run in order to gain information about their nature. The overall detection process is summarized in Fig. 1.15, where the processes that require the use of the 3D thermal model are indicated with an ellipse. The detection, quasi inverse and full-inverse procedures are based on the solution of the heat equation for different soil configurations. As explained, this is a very time consuming task that makes the whole algorithm inefficient for real on-field applications.

Target detection

The use of IR cameras taking images of the soil under inspection gives us the exact distribution of temperatures on the surface. On the other hand, the thermal model described previously and extensively validated with experimental data permits us to predict the thermal signature of the soil under given conditions, (López, 2003; López et al., 2004). The detection of the presence of potential targets on the soil is then made by comparing the measured IR images with the expected thermal behavior of the soil given by the solution of the forward problem under the assumption of absence of mines on the field, mathematically,

$$\alpha(x, y, z) = \alpha_{soil}, \qquad \forall x, y, z. \tag{1.29}$$

For this set of soil parameters, p, the application of the functional in Eq. (1.28) determines the surface positions (x, y) where the behavior is different from that expected under the assumption of mine absence, therefore revealing the presence of unexpected objects on the soil. These positions will be classified as potential targets, whereas the rest of the pixels (those that follow the expected pure-soil behavior) will be automatically classified as *soil*. This process is not trivial. The most straightforward approach, the thresholded detection,

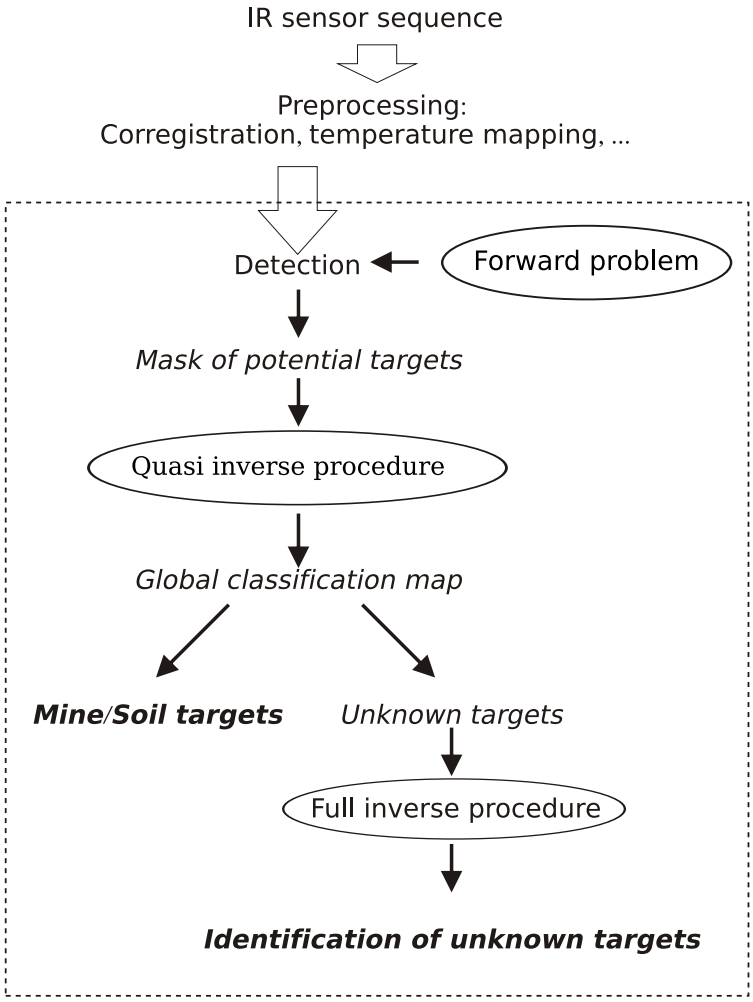
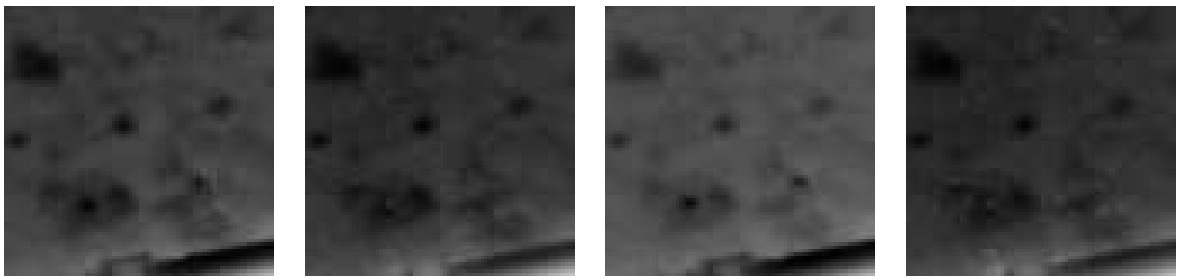


Figure 1.15: Structure of the approach used to detect buried landmines using infrared thermography.

has the drawback of setting the threshold, which will vary not only for different image sequences, but it is also likely to depend on the particular frame of the sequence, and on the characteristics of the measured data such as lighting conditions and the nature and duration of the heating. For this reason, the use of a reconfigurable structure, capable of adapting to varying experimental conditions was proposed on (López, 2003; López et al., 2004). In this work they demonstrated that it is possible to reduce the time frame of analysis to roughly one hour around sunrise as it is at this time when the maximum thermal contrast at the surface is expected. This phenomena can be better appreciated in Fig. 1.16, where a sequence of IR images of a mine field taken between 07:40 am and 08:40 am is shown. Taking into account the short time interval we can consider that



(a) 7:40 am (b) 08:00 am (c) 08:20 am (d) 08:40 am

Figure 1.16: Sample of measured IR images of a minefield at sunrise.

the properties of the soil remain unaltered and that there is no mass transference process during the simulation. The output of the detection stage is a black and white image with the mask of the potential targets.

Quasi inverse operator for the classification of the detected targets

In the previous section we dealt with the identification of the (x, y) position of the potential targets on the soil. In this section we will propose an operator for their classification into either *mine* or *soil* categories; any target that fails to fit into these categories will be classified as *unknown* (a procedure for the retrieval of further information about the nature of the *unknown* targets will be explained in the next section). For the *mine* category, the depth of burial will be also estimated. In general, this reconstruction is not possible unless additional information on the solution is incorporated in the model by means of the so-called *regularization techniques* (Kirsch, 1996; Engl et al., 1996). It is, however, possible to solve the inverse problem without the explicit use of a regularization strategy under proper initialization conditions and the use of iteration methods.

The iterative procedure is based on evaluating Eq. (1.28), which expresses the deviation between the observed IR data, y^δ, and the one given by the solution of the forward problem using known parameter distributions, $F[p]$. Therefore the heat equation needs to be solved for each of these distributions during the time of analysis (usually one hour around sunrise). In the case of mine targets, we will assume that their thermal evolution is driven by the thermal properties of the explosive used, which is commonly TNT composition B-3 or, less frequently, Tetryl. Our initial guess will be to assume that, *(i)* all the targets detected in

the detection step are mines, that is,

$$\alpha_{\text{target}} = \alpha_{\text{mine}}, \tag{1.30}$$

and *(ii)* the possible depths of burial constitute a discrete set $z \in \tilde{Z}$ being,

$$\tilde{Z} = \{k\,\Delta z, \quad \forall k = 0, 1, ..., d\}, \tag{1.31}$$

with Δz the discretization step and $d\,\Delta z$ the a depth of burial at which is satisfied the deep-ground condition, see Eq. (1.21). The situation $k = 0$ corresponds to surface-laid mines. These two assumptions imply a reduction of the search space, therefore the *quasi* inverse nature of the classification effort that will either confirm or reject them. Let,

- $\{y_s^\delta\}$, $s = 1, ..., S$, be the acquired IR image sequence, being S the total number of frames.
- $F[p]_{s,k}$, the modeled temperature distribution on the soil surface at time s. $F[p]_{s,k}$ is estimated by considering that all the detected targets are landmines buried at the depth given by index k in Eq. (1.31).

Note that, in the following, we will concentrate only on those areas of the image that were marked as possible targets in the detection phase. The classification map for the detected targets is obtained through the definition of a classification operator which includes the following computations:

1. For each time instant $s = 1, ..., S$ and burial depth $k = 0, ..., d$, an error map, $J_{s,k} = \|F[p]_{s,k} - y_s^\delta\|$, is estimated by evaluating Eq. (1.28) for each pixel position (x, y);
2. For each time instant s, a global error map (J_s) and a global classification map (Υ_s) are estimated iteratively by comparing the error maps $J_{s,k}$, $k = 0, ..., d$, as follows:
   - *Initialization step:* For each pixel (x, y), we set $J_s(x, y) = \varepsilon$ (where ε is a predefined threshold error value); and $\Upsilon_s(x, y) = soil$.
   - *Iterative update step:* For each depth of burial, k, with $k = 0, ..., d$, $J_s(x, y) = min(J_{s,k}(x, y), J_s(x, y))$ and $\Upsilon_s(x, y) = argmin_k(J_{s,k}(x, y), J_s(x, y))$ (the category for which the error is smaller, i.e. the depth of burial). If $J_s(x, y) > \varepsilon$ then $\Upsilon_s(x, y)$ is set to $Unknown$.

3. Once J_s and Υ_s have been obtained, we combine all these partial maps (J_s, resp. Υ_s) into single ones ($\mathbf{J}$, resp. $\boldsymbol{\Upsilon}$) in the following way:

   - $\boldsymbol{\Upsilon}$: Pixels classified as mines at any processing step are kept in the final classification map. For the others, we keep the category that appears more times.
   - $\mathbf{J}(x, y) = max_s(J_s(x, y))$. This is a very conservative approach aiming at reducing the number of false negatives (failure to detect a buried mine) even at the cost of increasing the false alarm rate of the system.
   - To find a tradeoff between the accuracy of the classification and the number of false alarms, we define a *cutoff error*, e_{max}. If the entry on the error map, $\mathbf{J}$ for a pixel exceeds e_{max}, the pixel will be automatically assigned to the category of *Unknown*. e_{max} is estimated empirically, however it could be estimated taking into account the pixels classified as non-mine based their temperature variance using bootstrap techniques, (Zoubir and Iskander, 2004).

Full-inverse procedure for the classification of non-mine targets

In this case, no assumption about the nature of the targets found in the detection phase is made, although the set of possible depths at which the targets can be placed is still bounded by Eq. (1.31). Under these assumptions, Eq. (1.30) does not hold and α_{target} is unknown and could take any value depending on the nature of the object. For this reason, it is necessary in this case to use a systematic approach for the minimization of the functional J, which implies the calculation of the gradient $\partial J/\partial p$.

Let us consider the existence of a buried target in a 3D soil volume, Ω, with an unknown $\alpha = \alpha(r)$, $r = (x, y, z) \in \Omega$. The thermal experiment is the following: at time $t = t_0$, the solid is subject to a prescribed flux, $q_{\text{net}}(r', t)$, on its surface Γ, being Γ the portion of the surface $\partial\Omega$ accessible for measurements. We then measure the temperature response $\theta(r', t)$ at the boundary $r' \in \Gamma$, during the time interval $[t_0, t_{\text{f}}]$. We rewrite our 3D forward problem in Eq. (1.18) as,

$$-\text{div}\{\alpha(r)\,\text{grad}\theta\} + \tfrac{\partial\theta}{\partial t} = 0, \quad r \in \Omega \tag{1.32a}$$

$$\theta(r, t = t_0) = \theta_0, \quad r \in \Omega \tag{1.32b}$$

$$\tfrac{\partial}{\partial n}\theta(r', t) = q_{\text{net}}(r', t) \quad r' \in \Gamma,\ t \in [t_0, t_{\text{f}}]. \tag{1.32c}$$

We look at the reconstruction of $\alpha(r)$ from the knowledge of the surface response of temperature, $y^\delta = \theta(r', t)$, to prescribed flux applied on the boundary $q_{\text{net}}(r', t)$. We call *data* the

pair $(\theta(r',t), q_{\text{net}}(r',t))$. As mentioned before, this is an ill-posed problem. It is intuitive that the data parameters (r',t) belong to a 3D subset, because $r' \in \Gamma$ and $t \in [t_0, t_{\text{f}}]$. This is sufficient enough for the reconstruction of the function $\alpha(r)$, defined in a 3D volume. Let us now introduce the *model* problem as an initial guess p, such that $p(r') = \alpha(r')$ (known data on the boundary), with the following governing equations and boundary conditions,

$$-\text{div}\{p(r)\,\text{grad}u\} + \tfrac{\partial u}{\partial t} = 0 \quad r \in \Omega \tag{1.33a}$$

$$u(r, t = t_0) = u_0, \quad r \in \Omega \tag{1.33b}$$

$$\tfrac{\partial}{\partial n} u(r,t) = q_{\text{net}}(r',t) \quad r' \in \Gamma, t \in [t_0, t_{\text{f}}]. \tag{1.33c}$$

The solution of Eq. (1.33) is a well-posed problem, as opposed to Eq. (1.32), and will be denoted by $u(r,t;p)$. Our aim will be to control p in such a way that the difference between the model and the observed data tends to zero. This goal is quantified by an objective function J to be minimized. The functional to be minimized is the L^2 norm of the misfit between the model and the observation given by,

$$J(u(p)) \equiv \frac{1}{2} \int_{t_0}^{t_{\text{f}}} \int_{\Gamma} \|u(r',t;p) - \theta(r',t)\|^2 dS\, dt. \tag{1.34}$$

This is a classic optimization problem which implies the calculation of the gradient of the functional J. To this aim we will make use of the variational method. If we introduce the notation,

$$< u, v >_{\Omega} = \int_{\Omega} u(r)\, v(r)\, d\Omega$$
$$< q_{\text{net}}, v >_{\Gamma} = \int_{\Gamma} q_{\text{net}}(r')\, v(r')\, dS$$
$$a_p < u, v > = \int_{\Omega} p\, \text{grad}u\, \text{grad}v\, d\Omega,$$

then the model problem, Eq. (1.33), is equivalent to the variational problem,

$$\int_{t_0}^{t_{\text{f}}} < \frac{\partial u}{\partial t}, v >_{\Omega}\ dt + \int_{t_0}^{t_{\text{f}}} (a_p < u, v >)\, dt - \int_{t_0}^{t_{\text{f}}} < q_{\text{net}}, v >_{\Gamma}\ dt = 0, \forall v \tag{1.35}$$

Eq. (1.35) can be considered as the constraints in the minimization problem, see Eq. (1.34). Therefore, we can introduce the Langrange multiplier $\lambda(r,t)$ and define the Lagrangian L as,

$$L(u,p,\lambda) \equiv J(u) + \int_{t_0}^{t_{\text{f}}} \{< \frac{\partial u}{\partial t}, \lambda >_{\Omega} + a_p < u, \lambda > - < q_{\text{net}}, \lambda >_{\Gamma}\}\, dt. \tag{1.36}$$

Note that $L = J$ if u is the solution of the model problem, Eq. (1.33), since Eq.(1.35) holds for any λ. Thus, the minimum of J under the constraints in Eq. (1.35) is the stationary

point of the Lagrangian L. Conversely, if $\delta L = 0$ for arbitrary $\delta\lambda$, u and p being held fixed, it follows necessarily that Eq. (1.35) holds. We consider,

$$\delta L = \frac{\partial L}{\partial u}\delta u + \frac{\partial L}{\partial p}\delta p, \tag{1.37}$$

where,

$$\tfrac{\partial L}{\partial u}\delta u \equiv \textstyle\int_{t_0}^{t_\mathrm{f}} (u-\theta, \partial u)_\Gamma \, dt + \int_{t_0}^{t_\mathrm{f}} \{(\delta\tfrac{\partial u}{\partial t}, \lambda)_\Omega + a_p(\delta u, \lambda)\}\, dt \tag{1.38a}$$

$$\tfrac{\partial L}{\partial p}\delta p \equiv \textstyle\int_{t_0}^{t_\mathrm{f}} \int_\Omega \delta p \,\mathrm{grad}u \,\mathrm{grad}\lambda \, d\Omega \, \delta\mu. \tag{1.38b}$$

We can restrict the choice of λ such that

$$\frac{\partial L}{\partial u}\delta u = 0. \tag{1.39}$$

This condition can be written as,

$$\int_{t_0}^{t_\mathrm{f}} < u-\theta, \delta u >_\Gamma \; dt + \int_{t_0}^{t_\mathrm{f}} \{- < \delta u, \frac{\partial \lambda}{\partial t} >_\Omega + a_p < \delta u, \lambda > \} \, dt + < \delta u, \lambda >_\Omega |_{t_0}^{t_\mathrm{f}} = 0, \tag{1.40}$$

where $\delta u(x, 0) = 0$. The last term of (1.40) vanishes if we impose,

$$\lambda(r, t \geq t_\mathrm{f}) = 0. \tag{1.41}$$

By doing so we obtain the equation for the adjoint field λ,

$$-\mathrm{div}\{p \,\mathrm{grad}\lambda\} - \tfrac{\partial \lambda}{\partial t} = 0, \quad r \in \Omega \tag{1.42a}$$

$$\lambda(r, t \geq t_\mathrm{f}) = 0, \quad r \in \Omega \tag{1.42b}$$

$$\tfrac{\partial}{\partial n}\lambda(r', t) = \theta - u, \quad r \in \Gamma. \tag{1.42c}$$

This is the so called *back diffusion* equation for the adjoint field, and it is also a well posed problem. With this choice of the adjoint field $\lambda(r, t)$, the variation ∂J becomes

$$\partial J = \int_{t_0}^{t_\mathrm{f}} \int_\Omega \partial p \,\mathrm{grad}u \,\mathrm{grad}\lambda \, d\Omega \, dt. \tag{1.43}$$

It results from Eq. (1.43) that the derivative of J in the $p(r)$ direction is known explicitly by solving two problems, the direct problem for the field u and the adjoint problem for the field λ. That is, the Z-integral,

$$\frac{\partial J}{\partial p} \equiv Z = \int_{t_0}^{t_\mathrm{f}} \mathrm{grad}u \,\mathrm{grad}\lambda \, dt. \tag{1.44}$$

Solving Eq. (1.33) and Eq. (1.42), both of them well-posed forward problems, and using Eq. (1.44), the expression of the update of Eq. (1.25)) can be calculated in a straightforward manner. With respect to the number of iterations of the Landweber method, the selection of the stopping criteria of the algorithm must be made according to the discrepancy principle in Eq. (1.26)). The bigger the η, the lower the number of iterations is, and the higher the error is. The selection of η for a particular application must then be a trade-off between computational time and accuracy of the solution.

The whole detection system was completely validated in (López, 2003) with real experiments, showing very good results in the detection of buried landmines.

Estimation of the computational cost of the non-destructive soil inspection procedure

The algorithms described in the previous sections are based on iterative procedures involving multiple solutions of the heat equation for different soil configurations. This constitutes a time consuming process which makes the full NDE procedure not feasible for its use on the field. As an example we will consider the analysis of a piece of soil of moderate dimensions of 1m×1m. As explained, the scope of applicability of IR techniques for mine detection is restricted to a depth of barely 10-15 cm, but the depth of analysis must be set to at least 40-50 cm in order to apply the boundary condition at the bottom surface in (1.21), which assumes that the temperature at a certain depth can be assumed to be constant. Using a uniform spatial discretization of $\Delta x = \Delta y = \Delta z = 0.8$ cm and assuming a temporal discretization step of $\Delta t = 6.25$ s, the simulation of the behavior of the soil during one hour using C++ on a Pentium IV 3GHz takes 5 minutes if single precision arithmetic is used to represent the temperatures (20 minutes if double precision is used). Taking into account that the proposed inverse procedures require the solution of the forward model for multiple soil configurations, the total computing assuming that only 100 iterations are needed (a soft approach) will add up to 8 hours (32 hours in double precision). This time is excessive for on the field experiments and, for this reason, two ways of reducing the computing time will be studied. The first one is a hardware heat equation solver accelerator. In this way the processes involving the use of the 3D thermal model run on this accelerator, thus reducing the computing time. This is indicated in Fig. 1.17, where the processes that makes use of the thermal model, executed on the hardware accelerator, are indicated with dashed lines. Another way of reducing the computing time is using non-uniform grids, which allows to decrease the spatial resolution, and therefore the number of grid points, in regions where

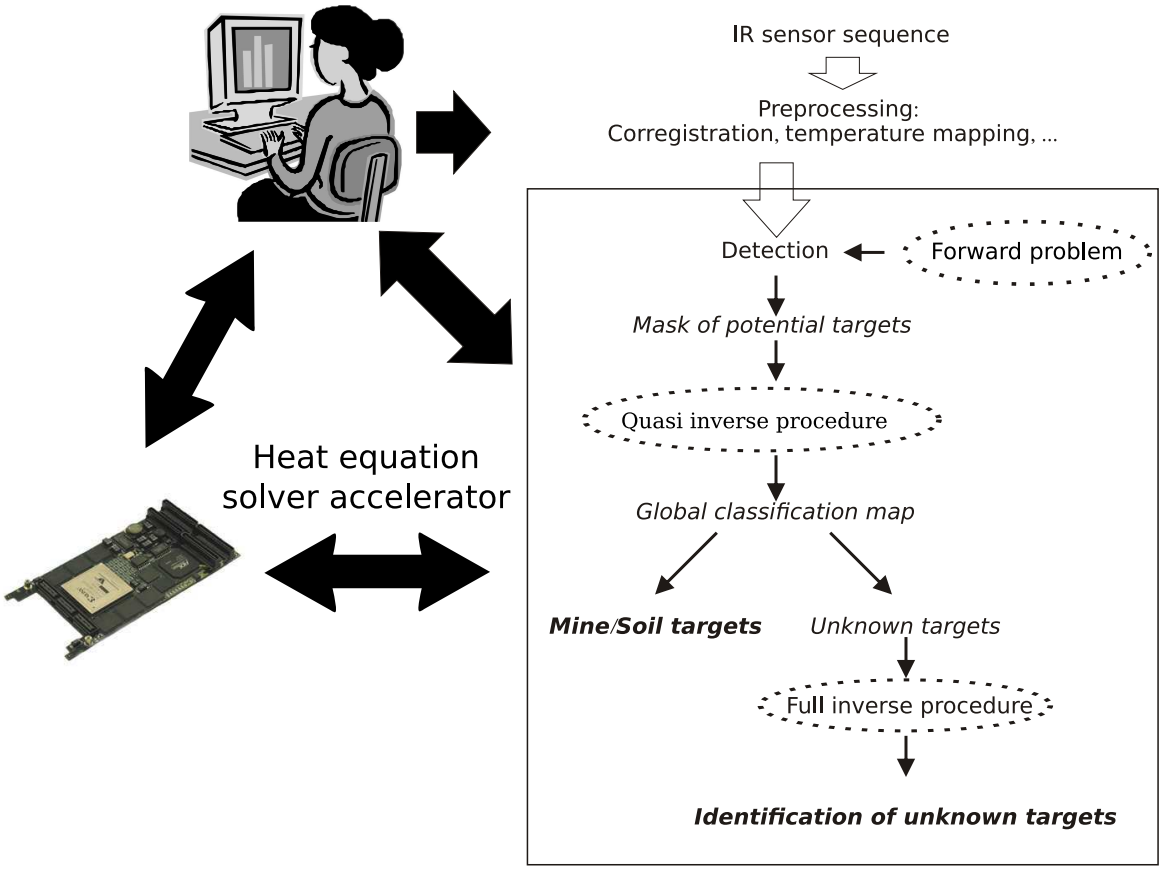


Figure 1.17: Structure of the algorithm used to detect buried landmines using infrared thermography.

the temperature varies slowly.

1.5 Summary

In this chapter we have introduced several NDE techniques focusing on those used for the detection of antipersonnel landmines. Then an infrared-thermography-based system for landmine detection was shown. This system is based on a thermal model of the soil; its validity, constituting the basis of the NDE procedure considered here, was thoroughly demonstrated in a previous paper, where several experimental setups were considered and excellent agreement between measured data and those given by the thermal model (implemented in MATLAB and C++) was shown, (López, 2003; López et al., 2004). The NDE procedure for the detection of landmines presented in that work is based on the realization

of the fact that the soil and the mines behave differently in response to a given thermal stimulus and that this is manifested as a thermal contrast on the soil surface. For this reason, and under natural heating conditions, sunrise and sunset are the times of the day at which this thermal contrast is largest. In fact, in the aforementioned work the authors have shown that a time interval of one hour around sunrise was enough to guarantee the correct detection of landmines on the soil.

The inverse engineering process proposed there to detect and classify buried landmines is based on the use of four IR images acquired with a 20 minutes interval from each other around sunrise which are compared to the thermal distribution given by the solution of the model for multiple soil/mine configurations in the same period. Taking into account that the inverse procedure requires the solution of the model for multiple soil configurations, the total computing time assuming that only 100 iterations are needed results on an computing time which makes the whole procedure unsuitable for on-field work applications. Thus, in this work we propose two techniques to reduce the computing time. The first one is an FPGA implementation of the thermal model that speeds up the computations making the system suitable for field work applications. A scheme of the algorithm once the FPGA system has been introduced can be seen in Fig. 1.17, where the processes makes use of the thermal model, running on the FPGA, are indicated with dashed points. The second approach consists on using non-uniform grids to reduce the number of nodes involved in the computations and therefore, the computing time.

Chapter 2

FD-TD solvers

2.1 Introduction

The FD-TD method is a widely used technique to solve PDE equations numerically. The FD-TD method has been used in order to express the set of equations describing physical phenomena in a discrete way and it has been successfully used in a variety of fields, most notably on electromagnetic simulations, but its applicability has been traditionally limited due to its demanding requirements in terms of memory and computing power, (Taflove, 1995). This limitation can be, however, avoided by means of a hardware implementation.

In this chapter we will introduce the FD-TD method, showing its main applications in PDE solving. After this, a review of the developed hardware implementations is done. Finally, the FD-TD method applied to the simulation of the thermal behavior is introduced.

2.2 FD-TD method

The FD-TD method transforms a PDE into a set of algebraic equations which allows to solve the system numerically. A generic PDE can be expressed as:

$$F(u(x,t)) = 0 \tag{2.1}$$

where F is a differential operator which includes time and space derivatives and u is a function of time and space. To solve numerically the PDE, spatial and temporal discretizations

are performed, obtaining a grid of points in the spatial domain and a set of times where the computations are performed. Two different numerical methods can be used: explicit and implicit methods. In the explicit method, the solution at a given point at time *m+1* depends only on the solution at neighboring points at time *m*. On the other hand, in the implicit method, the solution at a given point at time *m+1* depends on the solution at neighboring points at time *m+1* as well as the solution at time *m*. The advantage of the explicit methods is that a set of independent algebraic equations for each point of the grid is obtained; however to satisfy the numerical stability the time between each computation must be small. The implicit method is unconditionally stable and therefore the computations can be separated a large amount of time. However, the drawback of the implicit method is that at each time all node equations must be solved simultaneously, rather than sequentially (explicit method). We will focus on the explicit method because a hardware implementation of this method is more feasible for large size volume problems.

In the explicit method the PDE equation is discretized following the approach given by:

$$\frac{\partial u}{\partial x} \simeq \frac{u_{i+1} - u_i}{\Delta x} \tag{2.2}$$

$$\frac{\partial u}{\partial t} \simeq \frac{u^{m+1} - u^m}{\Delta t} \tag{2.3}$$

where the subscript i makes reference to the spatial discretization and the superscript m makes reference to the temporal discretization; and Δx and Δt are, respectively, the spatial and temporal discretization steps. This technique was introduced in (Yee, 1966) to solve numerically Maxwell's equations. As this method is widely used in the field of electromagnetic simulations, we will illustrate how it works using Maxwell's equations.

Maxwell's equations in an non-dispersive isotropic medium are, (Stratton, 1941):

$$\frac{\partial \overrightarrow{B}}{\partial t} + \nabla \times \overrightarrow{E} = 0 \tag{2.4}$$

$$\frac{\partial \overrightarrow{D}}{\partial t} - \nabla \times \overrightarrow{H} + \overrightarrow{J} = 0 \tag{2.5}$$

$$\nabla \cdot \overrightarrow{D} = \rho_v \tag{2.6}$$

$$\nabla \cdot \overrightarrow{B} = 0 \tag{2.7}$$

where $\overrightarrow{J}$[A/m^2] is the free current density, $\overrightarrow{E}$[V/m] is the electric field, $\overrightarrow{B}$[T] is the magnetic field, $\overrightarrow{H}$[A/m] is the magnetic field strength, $\overrightarrow{D}$[C/m^2] is the electric displacement

field and $\rho_v[\mathrm{C/m^3}]$ is the free electric charge density. In a rectangular coordinate system, see Fig. 2.1, Eqs. (2.4)- (2.5) are equivalent to:

$$-\frac{\partial B_x}{\partial t} = \frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z} \tag{2.8}$$

$$-\frac{\partial B_y}{\partial t} = \frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x} \tag{2.9}$$

$$-\frac{\partial B_z}{\partial t} = \frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} \tag{2.10}$$

$$\frac{\partial D_x}{\partial t} = \frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z} - J_x \tag{2.11}$$

$$\frac{\partial D_y}{\partial t} = \frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x} - J_y \tag{2.12}$$

$$\frac{\partial D_z}{\partial t} = \frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} - J_z \tag{2.13}$$

The FD-TD method provides a direct solution of this set of equations by discretizing space and time on uniform grids. The space and time derivatives are approximated by the central difference approximation. Fig. 2.1 shows the 3D grid called the *Yee-Cell*, (Yee, 1966). The *Yee-Cell* is a small cube of dimensions $\Delta x,\ \Delta y,\ \Delta z$, each point of the grid *(i,j,k)* has as real coordinates *($i\Delta x$,$j\Delta y$,$k\Delta z$)* in the real space. Instead of having three electric field components and three magnetic fields components in the center of each cell, the magnetic and electric fields components are interlaced. Thus, every electric field component is surrounded by four circulating magnetic field components and vice versa. The magnetic and electric field components are also interlaced in time; thus, the electric field components are updated at time $n\Delta t$, while the magnetic field components are updated at time $(n+\frac{1}{2})\Delta t$ using the electric field components obtained in $n\Delta t$. Applying the spatial and temporal discretizations, Eq. (2.8) is transformed into:

$$\begin{aligned}\frac{B_x^{n+1/2}(i,j+\frac{1}{2},k+\frac{1}{2}) - B_x^{n-1/2}(i,j+\frac{1}{2},k+\frac{1}{2})}{\Delta t} = \\ \frac{E_y^n(i,j+\frac{1}{2},k+1) - E_y^n(i,j+\frac{1}{2},k)}{\Delta z} \\ -\frac{E_z^n(i,j+1,k+\frac{1}{2}) - E_z^n(i,j,k+\frac{1}{2})}{\Delta y}\end{aligned} \tag{2.14}$$

The FD-TD equations corresponding to Eqs. (2.9)- (2.10) can be similarly constructed.

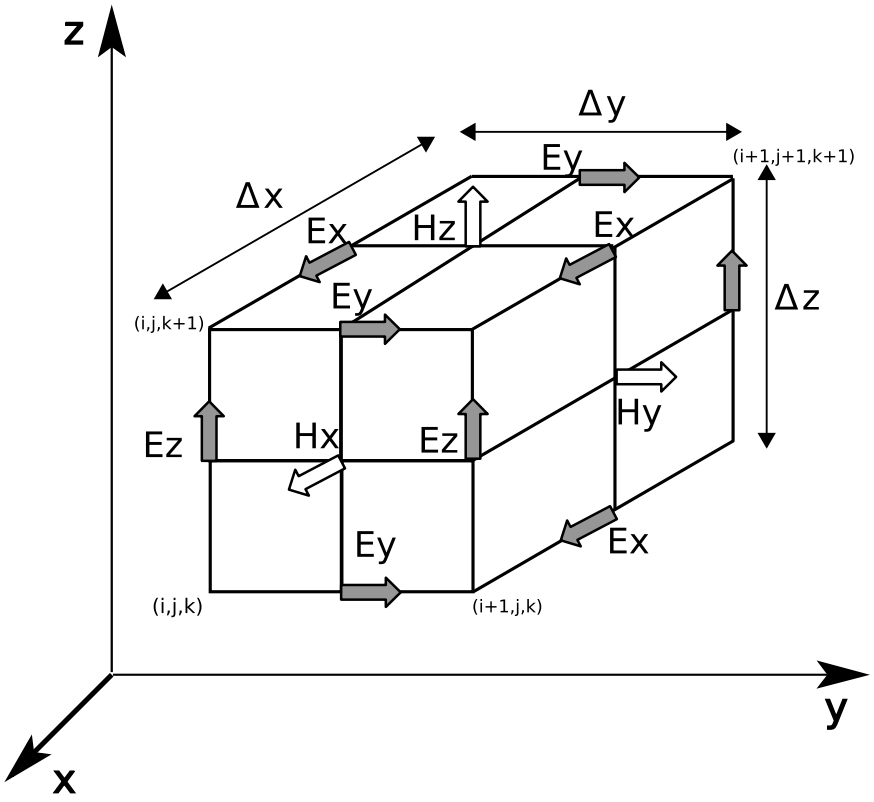


Figure 2.1: Geometrical representation of the Yee cell.

For Eq. (2.11) we obtain:

$$\begin{aligned} \frac{D_x^n(i+\frac{1}{2},j,k) - D_x^{n-1}(i+\frac{1}{2},j,k)}{\Delta t} = \\ \frac{H_z^{n-1/2}(i+\frac{1}{2},j+\frac{1}{2},k) - H_z^{n-1/2}(i+\frac{1}{2},j-\frac{1}{2},k)}{\Delta y} \\ -\frac{H_y^{n-1/2}(i+\frac{1}{2},j,k+\frac{1}{2}) - H_y^{n-1/2}(i+\frac{1}{2},j,k-\frac{1}{2})}{\Delta z} + J_x^{n-1/2}(i+\frac{1}{2},j,k) \end{aligned} \tag{2.15}$$

The FD-TD equations corresponding to Eqs. (2.12)- (2.13) can be similarly constructed. Thus, as can be noted, a set of algebraic equations is obtained, which includes additions and multiplications; this makes hardware implementation of a FD-TD solver suitable.

Using the FD-TD method a continuous PDE is translated into a set of algebraic equations; in the spatial domain this is reflected in the creation of a grid of discrete nodes that approximates the continuous volume; while, in the spatial domain it is reflected in the computations on discrete time intervals.

2.3 Review of FD-TD hardware implementations

In this section we will make a review of the different proposals of FD-TD hardware solvers and a comparison between all of them. All these hardware solvers implement Maxwell's equations. Our implementation is the first focused on the solution of the heat equation by means of a FD-TD hardware solver.

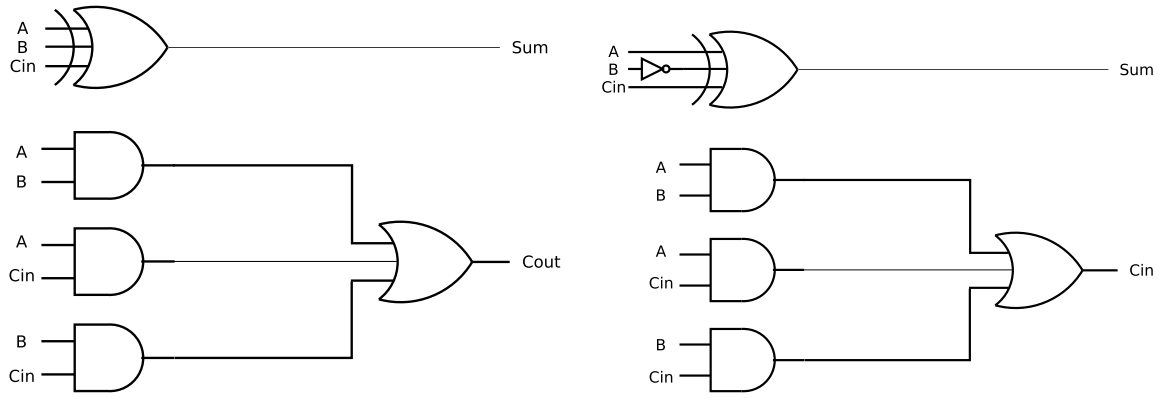
2.3.1 Schneider et al., 2002

The first hardware implementation of the FD-TD method to solve Maxwell's equations was reported in (Schneider et al., 2002a,b,c, 2004). In this work a Xilinx Virtex-XCV300, PQ240 package, was used to implement the FD-TD method and the FD-TD equations were solved using a pipelined bit-serial arithmetic by several reasons:

- The hardware cost is low. Adders, subtractors and memory elements are reused for each bit of the system wordlength.
- The size of the hardware that deals with each point of the grid is low, allowing to implement several nodes in parallel.
- The use of bit-serial arithmetic reduces the routing resources.

In addition, they chose integer arithmetic instead of floating arithmetic to save area resources on the FPGA. The implementation of the adders and subtractors can be seen in Fig. 2.2. The multiplier block was chosen from (Hartley and Parhi, 1995) and it implements one parallel N-bit coefficient and one serial multiplicand.

This design runs at 37.7 MHz and a new value is obtained every 849 ns. Each cell occupies 86.5 slices, and the full 1-D system (10 cells) occupies 30 % of the FPGA. They estimated that in a 2D system each cell would occupy 131 slices and that a 3-D system would occupy 256 slices. In this implementation the computation speed is independent of the number of simulated cells. However, this approach is limited to small size simulations due to the hardware limitation in the number of cells that fit on the FPGA. The authors proposed as future work a system with a few cells that process all elements acting as a pipeline. This implementation achieves a speedup of 10 compared to a PC-software solution.



(a) Adder (b) Subtractor

Figure 2.2: One bit adder and subtractor used to perform the N-bit serial adder

2.3.2 Placidi et al., 2002

This work presents a custom VLSI chip that implements the FD-TD method to solve Maxwell's equations using floating point arithmetic, (Placidi et al., 2002; Verducci et al., 2003). The system acts as an accelerating card attached to a PC via the PCI bus. The architecture of the system can be seen in Fig. 2.3. The main components of the system are the Floating Point Unit (FPU), the internal registers banks and the control unit. To maximize the utilization of the FPU, the data has conveniently been organized on the SDRAM. The data flow between the FPU and the SDRAM is managed by the control unit. The whole system has been developed using Register Transfer Level (RTL) hardware description language, VHDL, (VHDL, 1987), thus, making it independent of the digital hardware. The authors simulated the system using SDRAM models provided by the SDRAM vendors. The accuracy of the hardware implementation was tested by making a comparison with MATLAB simulations. A simulation of $5 \times 5 \times 10$ cells was carried out, however the SDRAM allows to do a simulation with up to $80 \times 80 \times 128$ cells. A digital synthesis was performed for every block with UMC VST $0.18\mu m$ and UMC VST 0.25 μm using standard cell design style. They obtained a speedup factor of 112 compared to a MATLAB simulation and a speedup factor of 5 compared to a FORTRAN simulation.

2.3.3 Durbano et al., 2003

The FD-TD hardware implementation of these authors was addressed in (Durbano et al., 2003a,b) and it implements Maxwell's equations. The system consists of a standard PC

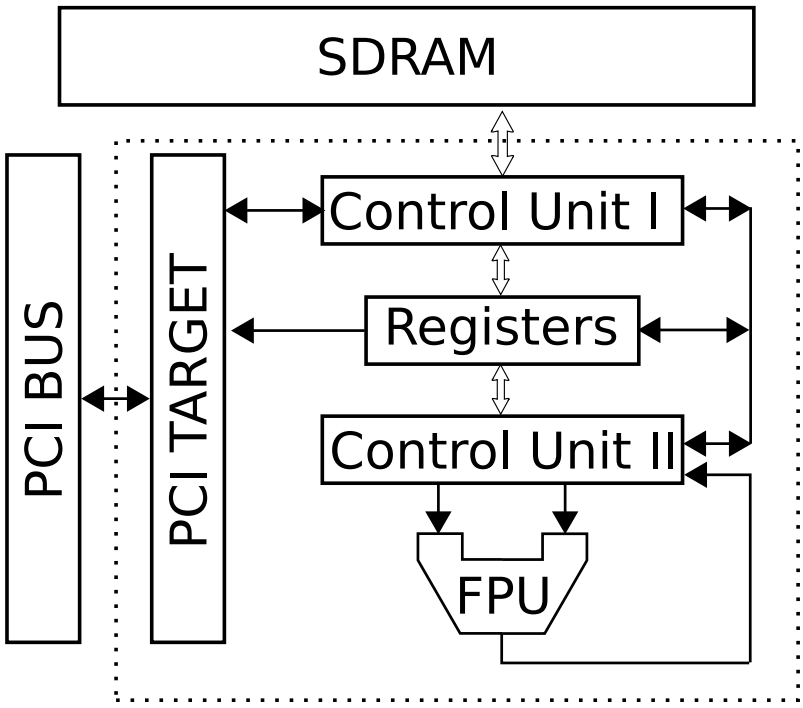


Figure 2.3: Architecture of the system proposed by Placidi et. al.

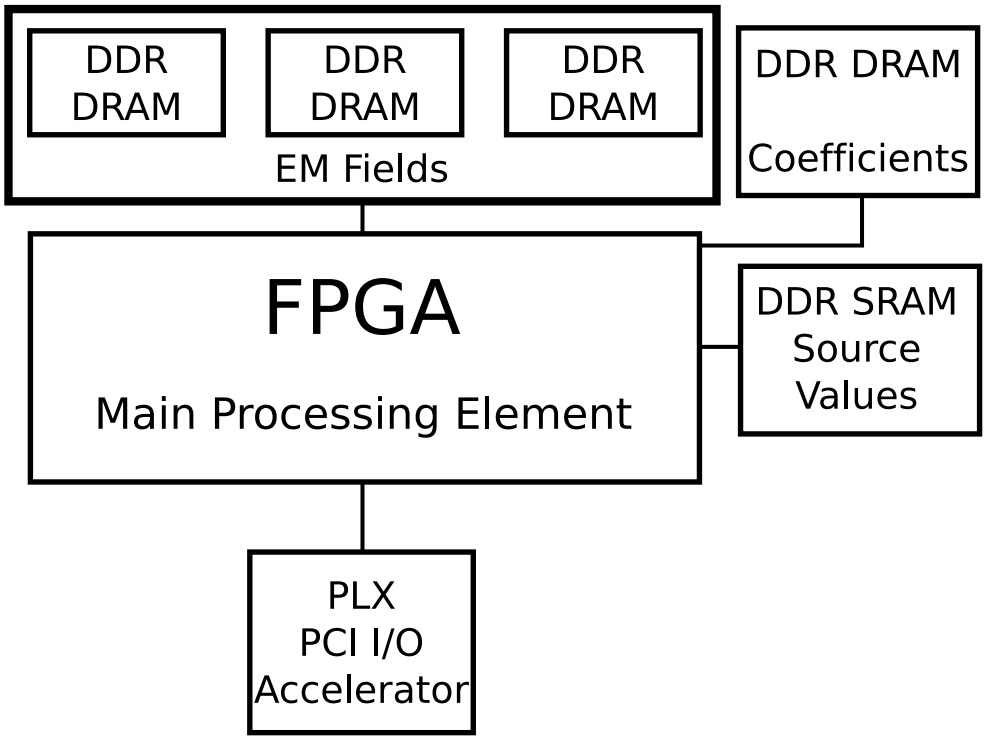


Figure 2.4: Block diagram of the hardware implementation proposed by Durbano et. al.

fitted with an off-the-self PCI-X card that holds a Xilinx Virtex-II 8000 FPGA, a PLX 9656 external PCI controller; 16 GB of DDR SDRAM and 36 Mb of DDR RAM, see Fig. 2.4. The architecture of the hardware implementation can be seen in Fig. 2.5. The system data flow begins with the counting and control unit (CCU), which provides the coordinates of the next electric or magnetic field to be updated. These coordinates are the inputs of the data dependence unit (DDU), which calculates the necessary data to update the field components. The coordinates from the DDU are passed to the RAM address decoder (RAD), that determines the memory location of the corresponding field components. The memory access is controlled by the memory switching unit (MSU) which manages all memory read/write processes. The data are stored in the three RAM banks and each bank contains the x, y or z field components and the material type of each node. The data read from the RAM is stored in the registers until it is necessary. Before the field updated is carried out several material coefficients must be computed; these coefficients are obtained in the material lookup table ($MLUT$). Finally, all the data are routed to the computation engine (CE) that performs the updating of the field components. Several CEs are included in the design, which allows to update several values in parallel.

The authors have used floating point arithmetic to avoid rounding errors, however the obtained design is not very efficient in terms of speed, (Durbano et al., 2003a). This first implementation was 9 times slower than a PC computing the FD-TD method. However, in further improvements of the implementation the authors have improved this result, achieving an speedup of 7 for small size simulations compared to a software solution (3 GHz Dell workstation with 2GB of PC3200 DDR SDRAM) and a speedup factor of 375 for large size simulations, when all data do not fit in the RAM of the PC and the access to the hard drive makes the PC-based solution slower, (Curt et al., 2007).

2.3.4 Chen et al., 2004

This work, (Chen et al., 2004), proposes the hardware implementation as a part of a NDE evaluation system. The system simulates the behavior of GPR and therefore it implements Maxwell's equations. The platform on which the system has been projected in a Firebird board from Annapolis Micro systems with 288 MB on-board memory; it uses a Xilinx Virtex-E xcv2000E FPGA. This implementation uses fixed-point arithmetic, where 26 bits are used to represent the magnetic and electric fields. The authors have done a complete analysis about the data quantization where they demonstrated the feasibility of using fixed

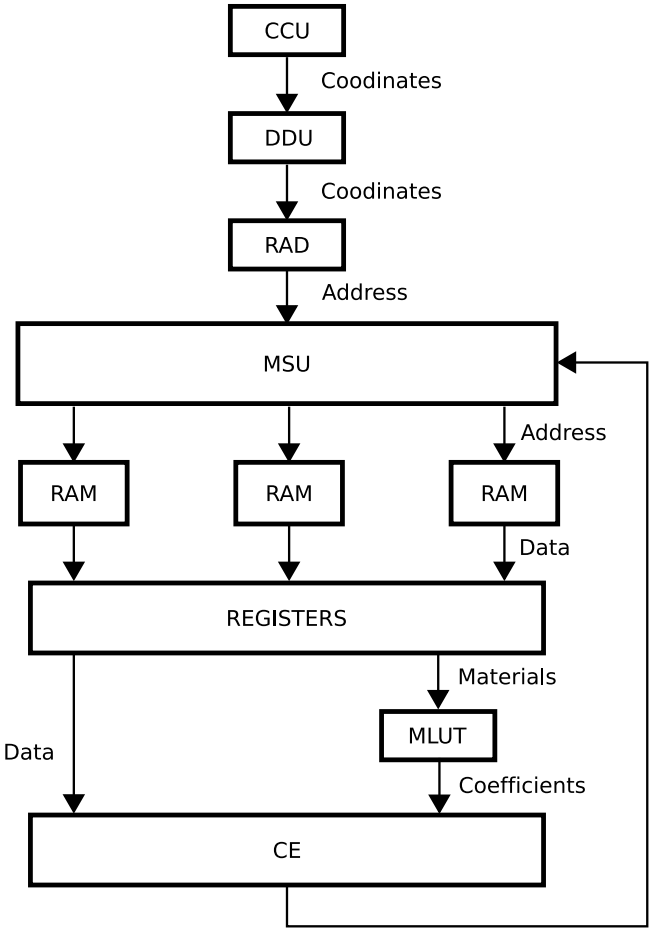


Figure 2.5: Data path of the architecture proposed by Durbano et. al.

point arithmetic, (Chen et al., 2004).

The data path of this implementation can be seen in Fig. 2.6. Each implemented cell has associated three electric and magnetic fields and several material properties parameters. The blockRAM memory interface module accesses the on-board memories and feeds the necessary data to the core updating pipelines. These pipelines are made up of two 30-bit multipliers and several adders and subtractors; the bottlenecks in these pipelines are the read/write process from memory and the multipliers. The multipliers have been pipelined to increase the frequency of the system and thus a multiplication takes 6 clock cycles. The first implementation of the system was limited to 2D simulations, however in (Chen et al., 2006) the authors have extended the model to 3D simulations. This implementation achieves a speedup factor of 24 compared to a PC software solution (3 GHz PC using FORTRAN).

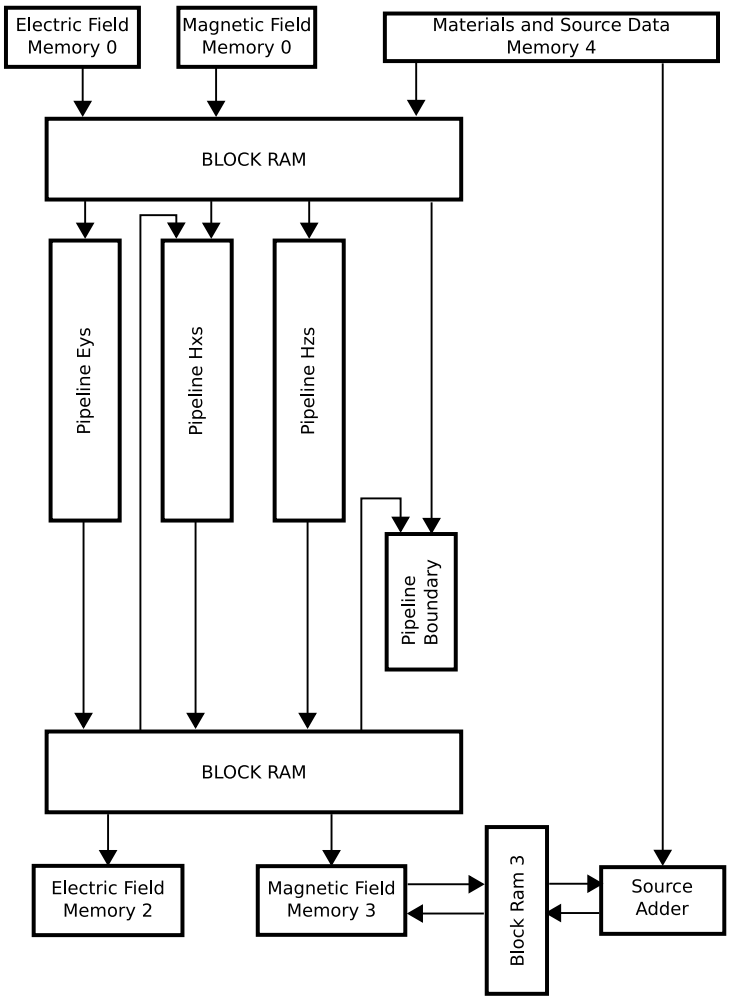


Figure 2.6: Hardware structure of Chen's FD-TD implementation.

2.3.5 Culley et al., 2005

In this work, (Culley et al., 2005), the authors have used a FPGA together with high performance computing (HPC) systems. They have targeted two reconfigurable HPC systems, both are Linux clusters with Xilinx FPGA acting as co-processors. The first one has 48 nodes, each of one has two Intel Pentium IV Xeon processors, 4GB of RAM and an Annapolis Micro Systems WildstarII PCI board, which includes two Xilinx Virtex-II FPGA (xc2v6000) and 12 MB of DDR II SRAM. The other HPC system has 18 nodes, each node has two 64-bit AMD Opteron processors and 4GB of RAM. Six of these nodes have a Virtex-II-Pro FPGA (xc2vp50-7) and four memory banks of 4MB each one. This approach uses both the FPGAs and the computational power of the HPC systems. The FD-TD com-

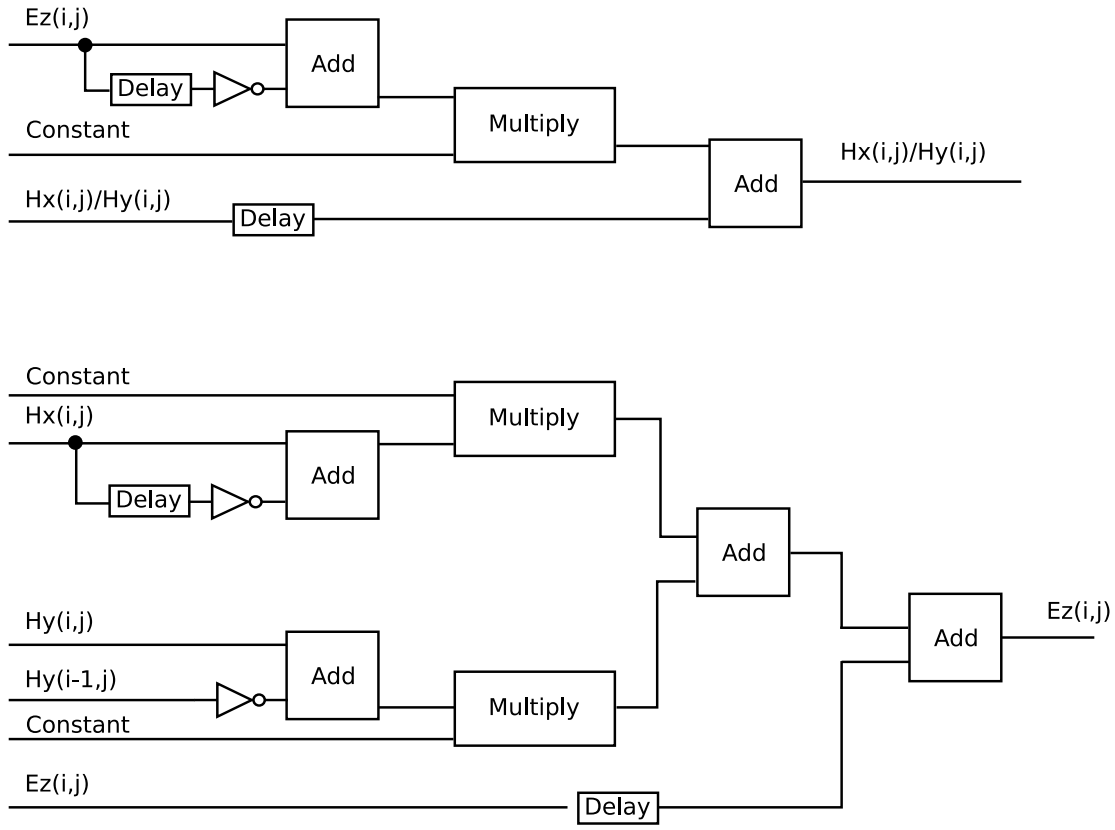


Figure 2.7: Block diagram of Culley's implementation to update H_x, H_y and H_z.

putation engine uses floating point modules, (Belanović and Leeser, 2002). The scheme of the hardware that implements the FD-TD calculations is shown in Fig. 2.7. The E_z update pipeline produces one output per clock and has a latency of 30 clock cycles; while the H_x and H_y pipelines take ones result per clock cycle and present a latency of 22 clock cycles. The authors double the throughput using FPGAs, compared to the computations using only the HPC.

2.3.6 He et al., 2005

Seismic modeling problems are governed by the acoustic wave or elastic wave equations. This work, (He et al., 2005, 2006), is focused in solving the acoustic wave equation in 2D space, (Gray, 2001):

$$\frac{\partial^2 P(x,y,z,t)}{\partial t^2} - v^2(x,y,z) \triangle P(x,y,z,t) = f(x,y,z,t) \tag{2.16}$$

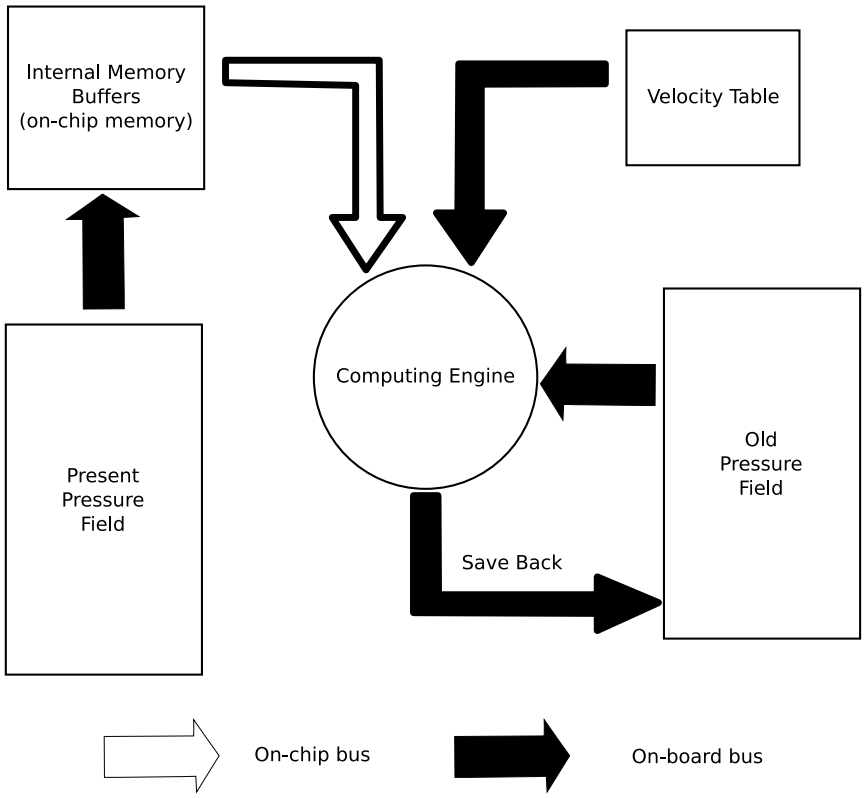


Figure 2.8: System architecture of He et al.

where P [Pa] is the pressure, v [m/s] is the acoustic velocity, f is the energy impulse and $\triangle \equiv \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial z^2}$ is the spatial Laplace operator. As can be noted, the structure is similar to that of Maxwell's equations. Due to the limited number of FPGA external pins, only few dedicated memory channels can be connected to the FPGA. Thus, the authors have developed an efficient on-chip memory architecture, which alleviates the memory bandwidth bottleneck with the external board memories. Based on the data dependency, when the updating is performed, they introduced a sliding window-based data buffering system between the computing engine and on-board external memory modules using FPGA blockRAM. Fig. 2.8 shows the architecture of the system, where we can see the on-board cache and how it provides the necessary data to the computing engine to perform the updating. The system is projected onto a Virtex-4 evaluation platform and they achieve a speedup factor up to 3 compared to a software solution. The authors proposed the introduction of more memory banks to increase the speedup, allowing to read more data in parallel.

2.3.7 Nagy et. al., 2006

This work, (Nagy et al., 2006), presents a cellular neural/non-linear network (CNN) based emulated digital architecture designed for the solution of PDE equations. The structure of the CNN paradigm, (Chua and Yang, 1988), makes it suitable to describe the behavior of systems with local connectivity. Thus, CNNs structure are perfect to solve PDE equations as their structure fits very well with the FD-TD method equations. The equations that give the evolution law for a discrete time CNN (DT-CNN) cell are:

$$x_{ij}(m+1) = \sum_{d \in k,l} A_{k,l} y^{k,l}(m) + \sum_{d \in k,l} B_{k,l} u^{k,l} + I \tag{2.17}$$

$$y_{i,j}(m+1) = f(x_{i,j}(m+1)) \tag{2.18}$$

where i, j are the indexes that makes reference to the particular cell of the array, x is the internal state of the cell, y is the output of the cell, u is the input of the cell, f is the activation function and I is a bias term. As can be noted, the DT-CNN equation that gives the evolution of a node can be adapted to those obtained in the FD-TD method. The authors have implemented the governing equations of the barotropic ocean model on a rotating Earth. The system was implemented on a RC200 board from Celoxica, (Celoxica, 2006), using 36 bits fixed-point arithmetic. Only one processing element fits on the FPGA and they simulated an array of 256x256 elements. In this implementation they achieve a speedup factor of 60 compared to a software-based solution.

2.3.8 Summary

In Table 2.1 the different hardware implementations to solve PDE equations are shown, where their main important characteristics are summarized. It can be noted that the implementations that use floating arithmetics achieve lower speedup than those that use integer arithmetics, except the implementation from (Durbano et al., 2003a) that achieves a high speedup when simulations of several millions of nodes is carried out. It also can be seen that almost all solvers implement Maxwell's equations.

2.4 FD-TD thermal model simulator

The FD-TD method allows to solve numerically a PDE where the continuous medium is replaced by an array of discrete points (nodes); this situation is illustrated in Fig. 2.9, where

Table 2.1: Summary of the different FD-TD hardware solvers.

Implementation	Hardware	Arithmetic	Implemented Eq.	Speed-up
Schneider et al., 2002	Xilinx Virtex-XCV300	Fixed point	Maxwell's equations	10
Placidi et al., 2002	Standard Cells	Floating point	Maxwell's equations	5
Chen et al., 2002	Firebird board	Fixed point	Maxwell's equations	30
Durbano et al., 2003	Custom FPGA board	Floating point	Maxwell's equations	7-375
Culley et al., 2005	WildstarII and Virtex-II-Pro FPGA	Floating point	Maxwell's equations	2
He et al., 2005	Xilinx Virtex-4	Floating point	Acoustic wave equation	3
Nagy et. al., 2006	RC200 Celoxica	Fixed point	Barotropic ocean equations	60

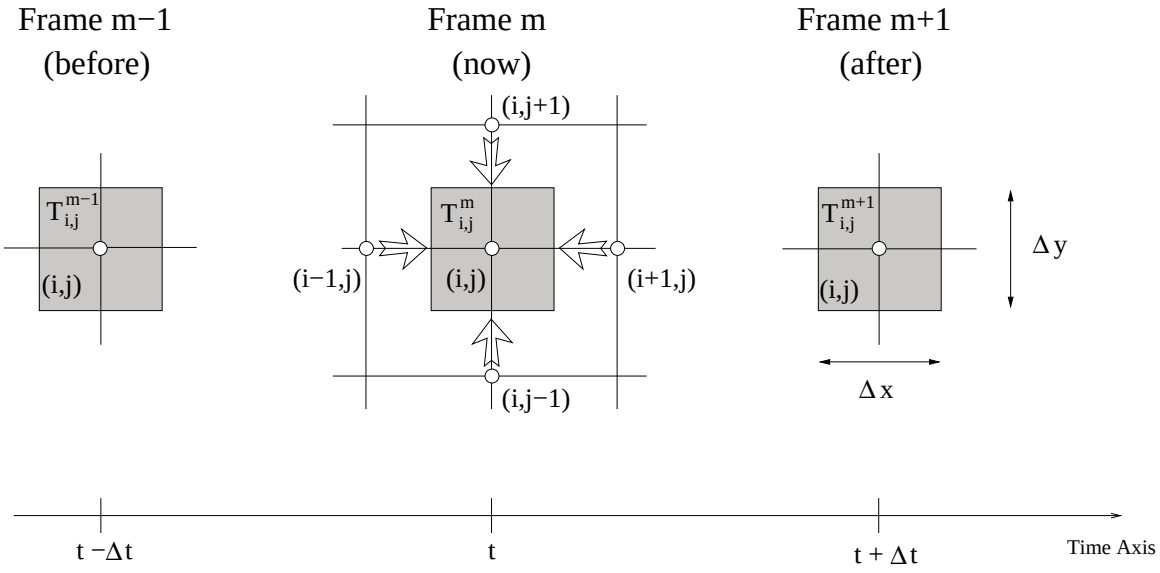


Figure 2.9: Two dimensional grid showing spatial and temporal discretization.

the spatial and temporal discretizations are shown for a 2D case. With this discretization each node represents a certain region and its temperature is a measure of the average temperature of the region. For example, the temperature of the node (i,j) in Fig. 2.9 may be viewed as the average temperature of the surrounding shared area. The numerical accuracy of the solution given by this method depends on the number of nodal points, thus, as the number of nodal points is increased a more accurate solution is obtained. As said previously, there are two schemes when applying the FD-TD method: explicit and implicit. A hardware implementation of the implicit method will be unaffordable for big volume size problems as all nodal equations must be solved in parallel. Thus, we introduce here the equations applying the explicit method to the heat equation.

The equations that describe the problem, Eqs. (1.18)-(1.22), are repeated here for the sake of clarity:

$$\frac{\partial T(\overrightarrow{r},t)}{\partial t} - div\,(\alpha(\overrightarrow{r})\ grad\,T(\overrightarrow{r},t)) = 0 \ \text{ with } \alpha = \frac{k}{\rho c_p} \tag{2.19}$$

$$-k\frac{\partial T(\overrightarrow{r},t)}{\partial n} = q_{net}(t) \quad for\ \Gamma \tag{2.20}$$

$$\frac{\partial T(\overrightarrow{r},t)}{\partial n} = 0 \quad for\ \partial\Omega\backslash\Gamma \tag{2.21}$$

$$T(\overrightarrow{r},t=t_0) = T_0(\mathbf{r}) \quad in\ \ \Omega \tag{2.22}$$

$$T(x,y,z\rightarrow\infty,t) = T_\infty \tag{2.23}$$

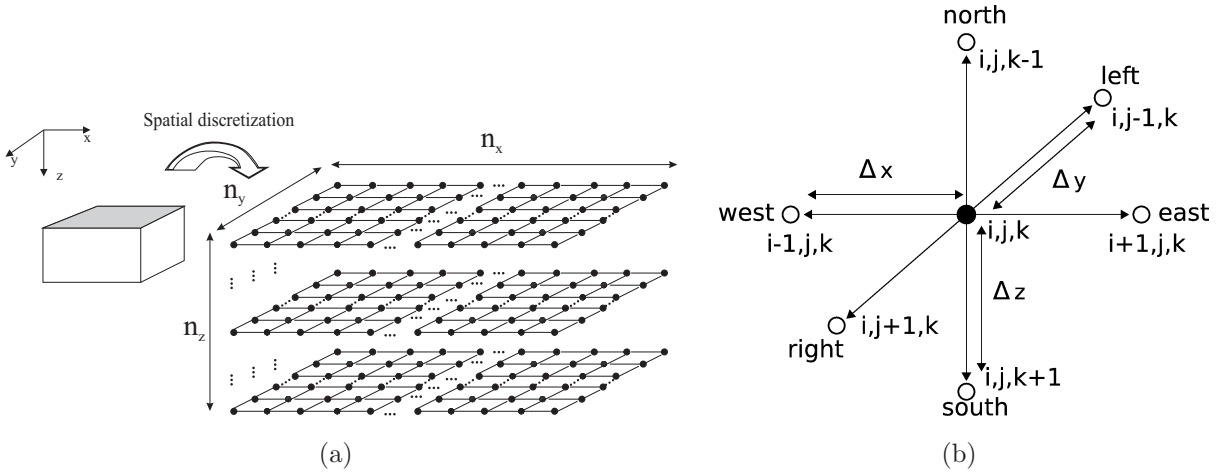


Figure 2.10: (a) Spatial discretization scheme of the continuous soil into a set of discrete nodes. (b) Considered neighbors to perform the updating process.

Now we will introduce the discretization of the thermal mode. Once the discretization is made, a grid that approximates the continuous medium is obtained. Fig. 2.10(a) shows this process and in Fig. 2.10(b) the considered neighboring of a node can be seen. The time derivative of the temperature can be expressed as

$$\frac{\partial T(\overrightarrow{r},t)}{\partial t} = \lim_{\Delta t \to 0} \frac{T(\overrightarrow{r},t+\Delta t) - T(\overrightarrow{r},t)}{\Delta t} \tag{2.24}$$

Thus, a good approximation for the time derivative at a point (i,j,k) of the grid is:

$$\frac{\partial T_{i,j,k}}{\partial t} \simeq \frac{T^{m+1}_{i,j,k} - T^{m}_{i,j,k}}{\Delta t} \tag{2.25}$$

where the m superscript makes reference to the time discretization steps and Δt is the temporal discretization step, i.e., the time interval between two iterations. Thus, the time derivative is expressed in terms of the temperature difference associated with the new $(m+1)$ and previous (m) times. In a similar way the spatial second derivatives can be approximated as:

$$\frac{\partial^2 T(\overrightarrow{r},t)}{\partial x^2} \simeq \frac{T_{i+1,j,k} + T_{i-1,j,k} - 2\,T_{i,j,k}}{\Delta x^2} \tag{2.26}$$

$$\frac{\partial^2 T(\overrightarrow{r},t)}{\partial y^2} \simeq \frac{T_{i,j+1,k} + T_{i,j-1,k} - 2\,T_{i,j,k}}{\Delta y^2} \tag{2.27}$$

$$\frac{\partial^2 T(\overrightarrow{r},t)}{\partial z^2} \simeq \frac{T_{i,j,k+1} + T_{i,j,k-1} - 2\,T_{i,j,k}}{\Delta z^2} \tag{2.28}$$

where Δx, Δy and Δz are the spatial discretization steps in the x, y and z directions, respectively. Hence, the computations are restricted to discrete points in time and space.

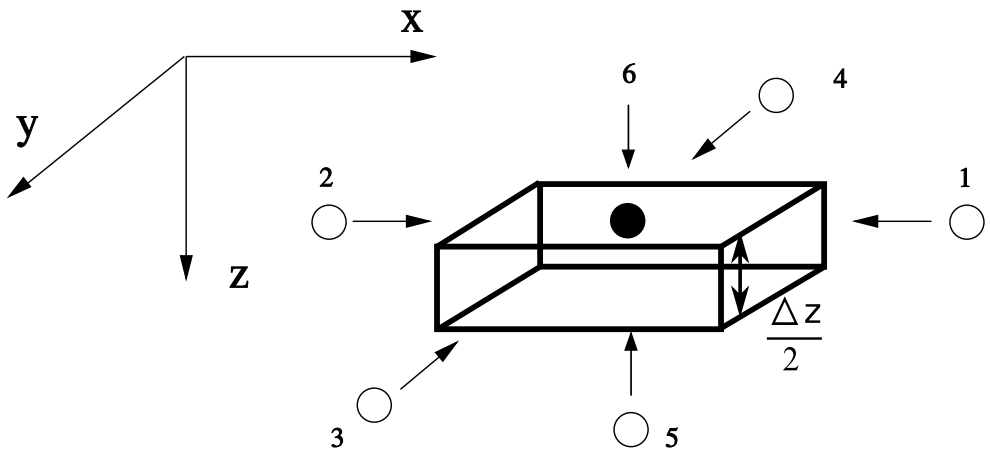


Figure 2.11: Heat flux contributions scheme from neighbor nodes for surface nodes.

2.4.1 Algebraic Equations

In order to solve the heat equation numerically, the discretized equations, Eqs. (2.25)-(2.28), are substituted in Eq.(2.19). For a 1D case the resulting equation is:

$$\frac{T_i^{m+1} - T_i^m}{\Delta t} = \alpha \frac{T_{i+1}^m + T_{i-1}^m - 2\,T_i^m}{(\Delta x)^2} \tag{2.29}$$

where α [m^2/s] is the thermal diffusivity and Δx and Δt are the spatial and temporal discretizations steps, respectively. The expression for a 3D system must include the influence of the external ambient conditions (radiation, convection). The expression can be obtained by applying the first law of the thermodynamics to each control volume:

$$\rho c_p \Delta x \Delta y \Delta z \frac{\partial T}{\partial t} = \sum_n q_n \tag{2.30}$$

where ρ [kg/m^3] is the density, c [J/Kg K] is the specific heat and q_n are the heat fluxes contributions from the neighbor nodes.

We will illustrate this process in boundary and surface nodes. Thus, the first law of thermodynamics for a surface node can be written as:

$$\rho c_p \Delta x \Delta y \frac{\Delta z}{2} \frac{\partial T}{\partial t} = \sum_n q_n \tag{2.31}$$

Fig. 2.11 shows the heat flux contributions from the neighbor nodes for a surface node. The node under consideration, the black one, receives a heat flux from the neighbor nodes,

labeled as 1-5, and from the external ambient conditions, labeled as 6. The equations for the heat fluxes that come from the different nodes can be written as:

$$q_1 = k\Delta y\frac{\Delta z}{2}\frac{T_{i+1,j,1}-T_{i,j,1}}{\Delta x} \quad q_2 = k\Delta y\frac{\Delta z}{2}\frac{T_{i-1,j,1}-T_{i,j,1}}{\Delta x} \tag{2.32}$$

$$q_3 = k\Delta x\frac{\Delta z}{2}\frac{T_{i,j+1,1}-T_{i,j,1}}{\Delta y} \quad q_4 = k\Delta x\frac{\Delta z}{2}\frac{T_{i,j-1,1}-T_{i,j,1}}{\Delta y} \tag{2.33}$$

$$q_5 = k\Delta x\Delta y\frac{T_{i,j,2}-T_{i,j,1}}{\Delta z} \tag{2.34}$$

$$q_6 = q_{net} = \Delta x\Delta y h(T_{air}-T_{i,j,1}) + \Delta x\Delta y 4\sigma\epsilon T_{air}^3(T_{air}-T_{i,j,1}) + \Delta x\Delta y\epsilon_{sun}q_{sun}^e \tag{2.35}$$

where h is the convective heat transfer coefficient, σ is the Stephan Boltzmann's constant, ϵ is the emissivity, T_{air} is the air temperature, ϵ_{sun} is the emissivity for the solar radiation, k is the thermal conductivity and q_{sun}^e is the measured solar radiation. Introducing these equations and the temporal discretization into the energy conservation equation, Eq. (2.31), we obtain the following equation that gives us the new temperature value for node as a function of the boundary conditions and the previous neighbor nodes temperature:

$$\begin{aligned} T_{i,j,1}^{m+1} =& (1-6F_0)T_{i,j,1}^m + 2\epsilon_{sun}F_0Sq_{sun}^m + 2F_0H(T_{air}-T_{i,j,1}^m) + \\ & F_0(T_{i+1,j,1}^m + T_{i-1,j,1}^m + T_{i,j+1,1}^m + T_{i,j-1,1}^m + 2T_{i,j,2}^m) + \\ & 8F_0RT_{air}^3(T_{air}-T_{i,j,1}^m) \end{aligned} \tag{2.36}$$

where we have assumed without loss of generality that $\Delta x = \Delta y = \Delta z$, and where:

$$F_0 = \frac{\alpha\Delta t}{(\Delta x)^2} \qquad R = \frac{\sigma\epsilon}{k}\Delta x \qquad S = \frac{\Delta x}{k} \qquad H = hS \tag{2.37}$$

For an internal node the conservation energy law can be written as:

$$\rho c\Delta x\Delta y\Delta z\frac{\partial T}{\partial t} = \sum_n q_n \tag{2.38}$$

and the heat flux contribution of each node, see Fig. 2.12, are:

$$q_1 = k\Delta y\Delta z\frac{T_{i+1,j,k}-T_{i,j,k}}{\Delta x} \quad q_2 = k\Delta x\Delta y\frac{T_{i-1,j,k}-T_{i,j,k}}{\Delta x} \tag{2.39}$$

$$q_3 = k\Delta x\Delta z\frac{T_{i,j+1,k}-T_{i,j,k}}{\Delta y} \quad q_4 = k\Delta x\Delta z\frac{T_{i-1,j,k}-T_{i,j,k}}{\Delta y} \tag{2.40}$$

$$q_5 = k\Delta x\Delta y\frac{T_{i,j,k+1}-T_{i,j,k}}{\Delta z} \quad q_6 = k\Delta x\Delta y\frac{T_{i,j,k-1}-T_{i,j,k}}{\Delta z} \tag{2.41}$$

Thus, the following equation is obtained:

$$T_{i,j,k}^{m+1} = (1-6F_0)T_{i,j,k}^m + F_0(T_{i+1,j,k}^m + T_{i-1,j,k}^m + T_{i,j+1,k}^m + T_{i,j-1,k}^m + T_{i,j,k+1}^m + T_{i,j,k-1}^m) \tag{2.42}$$

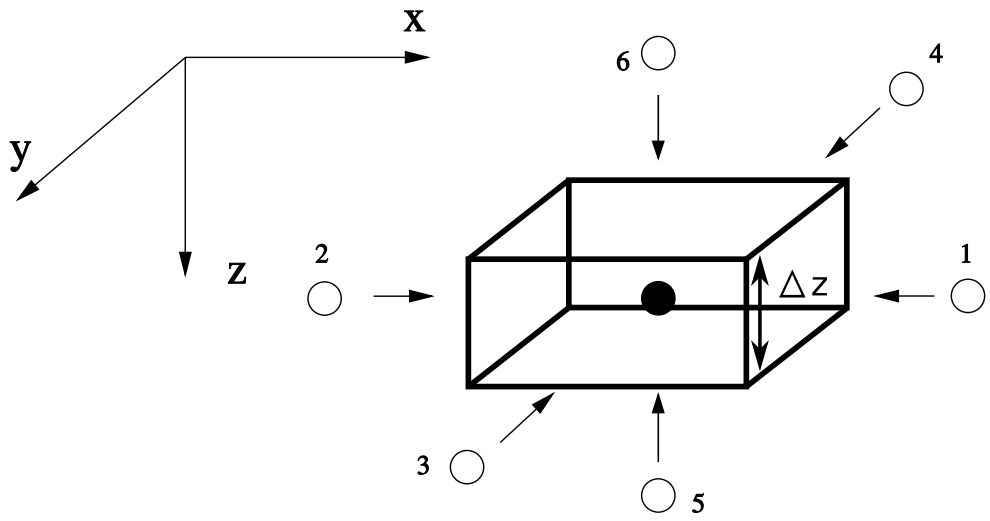


Figure 2.12: Heat flux contributions scheme from neighbor nodes for internal nodes.

Therefore we have obtained a set of equations that gives the node temperature evolution for both surface and internal nodes. To simulate the thermal behavior of the soil this set of equations is iteratively run until the desired simulation time has been reached.

The potential drawback of this method is that unless some precautions are taken the numerical solution can develop oscillations, whose amplitude increases from one time frame to the next. This numerical instability can be avoided with an appropriate selection of the spatial and temporal discretization steps. For an internal node the instability is avoided if the term $1 - 6F_0$ of Eq. (2.42) is non negative, which is translated into a small enough Δt, for a given Δx, that:

$$F_0 = \frac{\alpha \Delta t}{(\Delta x)^2} \leq \frac{1}{6} \tag{2.43}$$

Meanwhile for a surface node the stability condition is:

$$1 - 6F_0 - 2F_0 H - 8F_0 R T_{air}^3 \geq 0 \quad \Rightarrow \quad F_0 \leq \frac{1}{6 + 2H + 8RT_{air}^3} \tag{2.44}$$

In most situations:

$$2H + 8RT_{air}^3 << 6 \tag{2.45}$$

then Eq. (2.44) can be approximated as:

$$F_0 \leq \frac{1}{6 + 2H + 8RT_{air}^3} \approx \frac{1}{6} \tag{2.46}$$

As shown, the maximum Δt for a given Δx is limited by the condition on F_0. Eq. (2.44) defines a trade-off between the Δt and Δx parameters, known α, to fulfill the stability criteria.

In conclusion, this method is simply, each new time frame of numerical values $T_{i,j,k}^{m+1}$ is calculated in single-blow fashion by sweeping through the domain and using Eqs. (2.36) and (2.42). The drawback of this method is that it is conditionally stable, being the condition that F_0 must not exceeded a certain value. However this is not a problem in most situations.

Let us analyze now how well the FD-TD scheme approximates the continuous problem; the called *truncation error* (TE), that appears due to the discretization. To this aim we will use the Taylor series expansion. Using the Taylor expansion series and considering a one dimensional case, we can approximate T_i^{m+1} as:

$$T_i^{m+1} = T_i^m + \frac{\Delta t}{1!}\frac{\partial T_i^m}{\partial t} + \frac{(\Delta t)^2}{2!}\frac{\partial^2 T_i^m}{\partial t^2} + \dots \tag{2.47}$$

Thus:

$$\frac{T_i^{m+1} - T_i^m}{\Delta t} = \frac{\partial T_i^m}{\partial t} - \frac{1}{2}\Delta t\frac{\partial^2 T_i^m}{\partial t^2} + \mathcal{O}(\Delta t^2) \tag{2.48}$$

it should be noted that the term of order Δt can be large. In fact, solving problems where the solution has sharp changes with respect to time this term can be very large. But generally, if Δt is sufficiently small, we can ignore the term of order Δt and use Eq. (2.25) as a good approximation of time derivative. Using the same approximation we can obtain:

$$T_{i+1}^m = T_i^m + \frac{\Delta x}{1!}\frac{\partial T_i^m}{\partial x} + \frac{(\Delta x)^2}{2!}\frac{\partial^2 T_i^m}{\partial x^2} + \frac{(\Delta x)^3}{3!}\frac{\partial^3 T_i^m}{\partial x^3} + \frac{(\Delta x)^4}{4!}\frac{\partial^4 T_i^m}{\partial x^4} + \mathcal{O}(\Delta x^5) \tag{2.49}$$

$$T_{i-1}^m = T_i^m - \frac{\Delta x}{1!}\frac{\partial T_i^m}{\partial x} + \frac{(\Delta x)^2}{2!}\frac{\partial^2 T_i^m}{\partial x^2} - \frac{(\Delta x)^3}{3!}\frac{\partial^3 T_i^m}{\partial x^3} + \frac{(\Delta x)^4}{4!}\frac{\partial^4 T_i^m}{\partial x^4} + \mathcal{O}(\Delta x^5) \tag{2.50}$$

and

$$\frac{T_{i+1}^m - 2T_i^m + T_{i-1}^m}{\Delta x^2} = \frac{\partial^2 T_i^m}{\partial x^2} + \mathcal{O}(\Delta x^2) \tag{2.51}$$

Considering the continuous one dimensional heat equation and introducing the Taylor series expansion of the derivatives we see that:

$$\frac{\partial T_i^m}{\partial t} - \alpha\frac{\partial^2 T_i^m}{\partial x^2} = \frac{T_i^{m+1} - T_i^m}{\Delta t} - \alpha\frac{T_{i+1}^m - 2T_i^m + T_{i-1}^m}{\Delta x^2} + \mathcal{O}(\Delta t) + \mathcal{O}(\Delta x^2) \tag{2.52}$$

Thus, we see that the difference equations approximate the partial differential equation to the first order in Δt and to the second order in Δx. Using Eqs. (2.48)-(2.50) the TE can be expressed as:

$$TE = -\frac{1}{2}\Delta t\frac{\partial^2 T_i^m}{\partial t^2} + \frac{\alpha}{12}\frac{\partial^4 T_i^m}{\partial x^4}(\Delta x)^2 + \mathcal{O}(\Delta t^2) + \mathcal{O}(\Delta x^4) \tag{2.53}$$

Using the heat equation, the temporal derivative term of Eq. (2.53) can be rewritten as:

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2} \Rightarrow \frac{\partial^2 T}{\partial t^2} = \frac{\partial}{\partial t}\left(\alpha \frac{\partial^2 T}{\partial x^2}\right) = \alpha \frac{\partial^2}{\partial x^2}\left(\frac{\partial T}{\partial t}\right) = \alpha \frac{\partial^2}{\partial x^2}\left(\alpha \frac{\partial^2 T}{\partial x^2}\right) = \alpha^2 \frac{\partial^4 T}{\partial x^4} \quad (2.54)$$

Introducing this expression in Eq.(2.53), the truncation error can be written as:

$$TE = (1 - 6F_0)\frac{\alpha}{12}\frac{\partial^4 T}{\partial x^4}(\Delta x)^2 + \mathcal{O}(\Delta t^2) + \mathcal{O}(\Delta x^4) \quad (2.55)$$

Thus, if we chose $F_0 = 1/6$, then the truncation error is of second order in Δt and of fourth order in Δx, which gives a higher precision. However, the choice of $\Delta t = (\Delta x)^2/6\alpha$ implies time steps too small to be of practical interest in most cases, (Arpaci et al., 2000). As the spatial and temporal discretization steps tend to zero, the truncation error tends to zero and then the FD-TD set of equations tends to the continuous problem, which shows the consistency of the discrete formulation.

As has been illustrated, the numerical solution is expected to approach the exact solution. However, computer solutions may be limited in accuracy by the number of digits employed by the processor. This restriction associated with rounding to a finite number of digits results in *round-off errors*. These errors are increased with the number of arithmetic operations required to produce a solution, hence with the number of grid points The accuracy of a numerical solution is expected to be a trade-off between truncation errors and round-off errors.

2.4.2 Non-uniform media

All the equations shown until now correspond to the simulation of a homogeneous piece of soil. However we need to simulate a piece of soil with buried objects; this issue involves some modifications of the FD-TD equations, see (Rohsenow and Hartnett, 1973), to obtain a more realistic model. The situation that we are considering now is reflected in Fig. 2.13, for a 2-D case, where we have a layer that is the interface between two materials with different thermal properties, labeled as A and B. Considering the 3D case and applying the energy conservation law for an internal node we obtain:

$$\Delta x \Delta y \Delta z \left(\frac{\rho_A c_A}{2} + \frac{\rho_B c_B}{2}\right)\frac{\partial T}{\partial t} = \sum_n q_n \quad (2.56)$$

where q_n is the heat flux contribution of the neighbor nodes to the node under consideration, see Fig 2.12. The equations for the heat fluxes that come from the different nodes can be

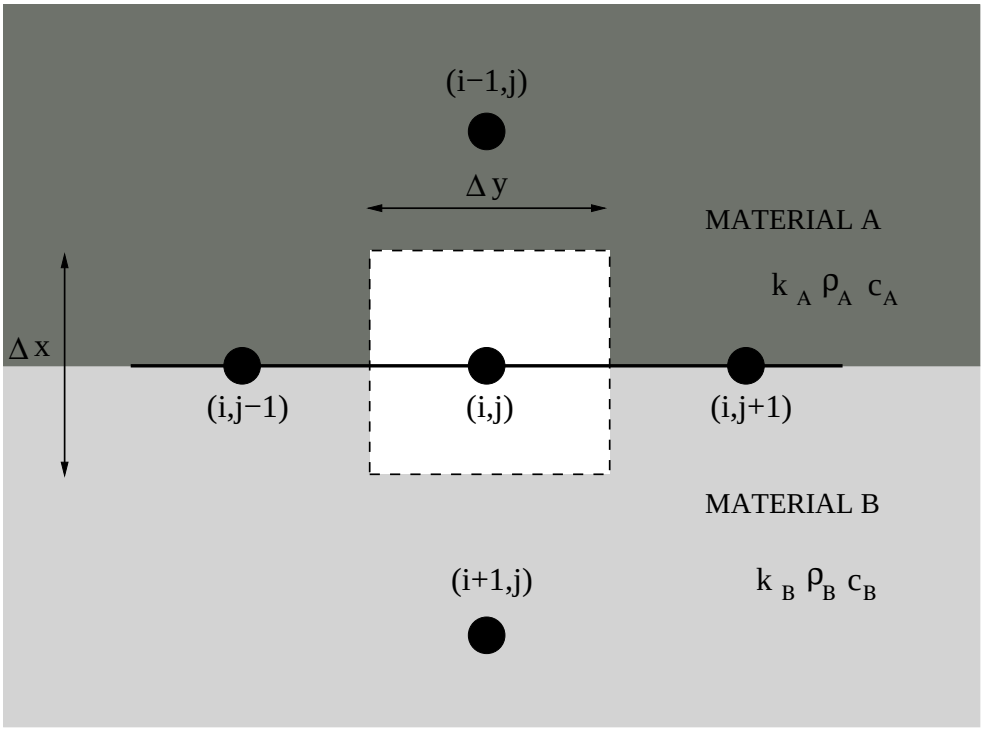


Figure 2.13: Heat conduction between adjoining and dissimilar materials (2D case).

written as:

$$q_1 = \frac{k_A + k_B}{2}\Delta y \Delta z \frac{T_{i+1,j,k} - T_{i,j,k}}{\Delta x} \quad q_2 = \frac{k_A + k_B}{2}\Delta y \Delta z \frac{T_{i-1,j,k} - T_{i,j,k}}{\Delta x} \tag{2.57}$$

$$q_3 = \frac{k_A + k_B}{2}\Delta x \Delta z \frac{T_{i,j+1,k} - T_{i,j,k}}{\Delta y} \quad q_4 = \frac{k_A + k_B}{2}\Delta x \Delta z \frac{T_{i,j-1,k} - T_{i,j,k}}{\Delta y} \tag{2.58}$$

$$q_5 = k_B \Delta x \Delta y \frac{T_{i,j,k+1} - T_{i,j,k}}{\Delta z} \quad q_6 = k_A \Delta x \Delta y \frac{T_{i,j,k-1} - T_{i,j,k}}{\Delta z} \tag{2.59}$$

If we introduce these terms in Eq. (2.56) we obtain the following equation, assuming without loss of generality, that $\Delta x = \Delta y = \Delta z$, for a node located at the interface between two materials with different thermal properties:

$$\begin{aligned} T_{i,j,k}^{m+1} =& T_{i,j,k}^{m} + \Delta t \frac{k_A + k_B}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} \frac{T_{i+1,j,k}^{m} + T_{i-1,j,k}^{m} + T_{i,j+1,k}^{m} + T_{i,j-1,k}^{m} - 4T_{i,j,k}^{m}}{(\Delta x)^2} + \\ & \frac{2k_A \Delta t}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} \frac{T_{i,j,k-1}^{m} - T_{i,j,k}^{m}}{(\Delta x)^2} + \frac{2k_B \Delta t}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} \frac{T_{i,j,k+1}^{m} - T_{i,j,k}^{m}}{(\Delta x)^2} \end{aligned} \tag{2.60}$$

This equation can be re-written as:

$$\begin{aligned} T_{i,j,k}^{m+1} =& T_{i,j,k}^{m} + F_{0xy}(T_{i+1,j,k}^{m} + T_{i-1,j,k}^{m} + T_{i,j+1,k}^{m} + T_{i,j-1,k}^{m} - 4T_{i,j,k}^{m}) + \\ & F_{0zA}(T_{i,j,k-1}^{m} - T_{i,j,k}^{m}) + F_{0zB}(T_{i,j,k+1}^{m} - T_{i,j,k}^{m}) \end{aligned} \tag{2.61}$$

where:

$$F_{0xy} = \frac{\Delta t}{(\Delta x)^2}\frac{k_A + k_B}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} \quad F_{0zA} = \frac{\Delta t}{(\Delta x)^2}\frac{2k_A}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} \quad F_{0zB} = \frac{\Delta t}{(\Delta x)^2}\frac{2k_B}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} \tag{2.62}$$

which has the same structure of Eq. (2.42), where the F_0 terms that multiplies to the z terms are different from those of the x and y terms. Note that if *Material A = Material B* then Eq. (2.62) is transformed into Eq. (2.42). Finally, the equation that gives the temperature of a surface node remains unaltered, see Eq. (2.36).

Now we will analyze the stability condition in this case. If we group the terms of Eq. (2.60) which multiply to $T^m_{i,j,k}$ we have the following stability condition:

$$1 - \frac{\Delta t}{(\Delta x)^2}\left(4\frac{k_A + k_B}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} + 2\frac{k_A}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} + 2\frac{k_B}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}}\right) > 0 \tag{2.63}$$

If $k_M = max(k_A, k_B)$, then:

$$1 - \frac{\Delta t}{(\Delta x)^2}\left(4\frac{k_A + k_B}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} + 2\frac{k_A}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} + 2\frac{k_B}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}}\right) > 1 - 12\frac{\Delta t}{(\Delta x)^2}\frac{k_M}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} \tag{2.64}$$

Finally, if $\left(\frac{k_m}{\alpha_m}\right) = min\left(\frac{k_A}{\alpha_A}, \frac{k_B}{\alpha_B}\right)$, then:

$$1 - 12\frac{\Delta t}{(\Delta x)^2}\frac{k_A}{\frac{k_A}{\alpha_A} + \frac{k_B}{\alpha_B}} > 1 - 6\frac{\Delta t}{(\Delta x)^2}\frac{k_M \alpha_m}{k_m} \tag{2.65}$$

and therefore the stability criteria can be written as follows:

$$F_0 = \frac{k_M}{k_m}\frac{\alpha_m \Delta t}{(\Delta x)^2} < \frac{1}{6} \tag{2.66}$$

2.4.3 Non-uniform grid

The considered spatial discretization made until now is based on an uniform grid, that is, the spatial discretization steps are always the same, independently of the position of the depth considered. However, a modification of the grid can be made taking into account the nature of the problem under analysis. In the heat transfer processes the layers near the surface are those that suffer the effects of the atmospheric conditions; hence the temperature in these layers varies quicker than at higher depths. Due to this fact the discretization could be more accurate in layers near the surface than at higher depths. Thus, a grid

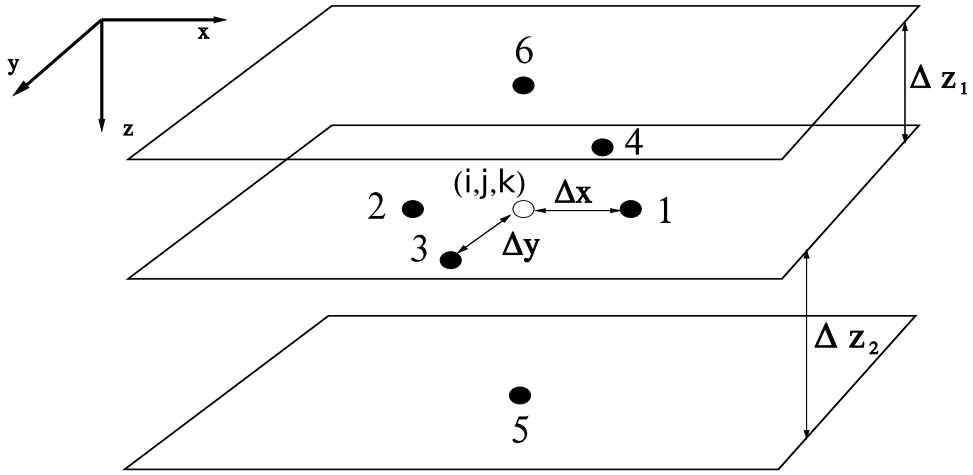


Figure 2.14: Scheme of a discretization with a non-uniform grid in the z direction.

with a small spatial discretization step on the layers near the surface and a bigger spatial discretization step as the depth increases is preferred. This scheme can be used to obtain two advantages; firstly a higher spatial resolution can be achieved in the regions where temperatures variations are higher or to represent small objects, such as mines, obtaining a higher precision; secondly this can be used to reduce the number of layers needed to simulate the system; thus reducing the computing time. Let us see how the equations are modified for a 3D case when the spatial discretization step in the z direction changes, see Fig. 2.14. In this case, applying the energy conservation law for an internal node we have:

$$\rho c \Delta x \Delta y \frac{\Delta z_1 + \Delta z_2}{2} \frac{\partial T}{\partial t} = \sum_n q_n \tag{2.67}$$

where q_n is the heat flux contribution of the neighbors nodes to the node under consideration. The equations for the heat fluxes that come from the different nodes can be written as:

$$q_1 = k \frac{\Delta z_1 + \Delta z_2}{2} \Delta y \frac{T_{i+1,j,k} - T_{i,j,k}}{\Delta x} \quad q_2 = k \frac{\Delta z_1 + \Delta z_2}{2} \Delta y \frac{T_{i-1,j,k} - T_{i,j,k}}{\Delta x} \tag{2.68}$$

$$q_3 = k \frac{\Delta z_1 + \Delta z_2}{2} \Delta x \frac{T_{i,j+1,k} - T_{i,j,k}}{\Delta y} \quad q_4 = k \frac{\Delta z_1 + \Delta z_2}{2} \Delta x \frac{T_{i,j-1,k} - T_{i,j,k}}{\Delta y} \tag{2.69}$$

$$q_5 = k \Delta x \Delta y \frac{T^m_{i,j,k+1} - T_{i,j,k}}{\Delta z_2} \quad q_6 = k \Delta x \Delta y \frac{T^m_{i,j,k-1} - T_{i,j,k}}{\Delta z_1} \tag{2.70}$$

Introducing these heat fluxes into Eq. (2.67) the equation that gives the temperature of an internal node is given:

$$\begin{aligned} T^{m+1}_{i,j,k} =& T^m_{i,j,k} + F_{0x}(T^m_{i-1,j,k} + T^m_{i+1,j,k} + T^m_{i,j-1,k} + T^m_{i,j+1,k} - 4T^m_{i,j,k}) \\ & F_{0z1}(T^m_{i,j,k-1} - T^m_{i,j,k}) + F_{0z2}(T^m_{i,j,k+1} - T_{i,j,k}) \end{aligned} \tag{2.71}$$

where we have assumed that $\Delta x = \Delta y \neq \Delta z_1 \neq \Delta z_2$ and where:

$$F_{0x} = \frac{\alpha \Delta t}{(\Delta x)^2} \quad F_{0z1} = \frac{2\alpha \Delta t}{\Delta z_1(\Delta z_1 + \Delta z_2)} \quad F_{0z2} = \frac{2\alpha \Delta t}{\Delta z_2(\Delta z_1 + \Delta z_2)} \tag{2.72}$$

For a surface node the q_6 term is the same of Eq. (2.35), and applying the conservation energy law we obtain:

$$\begin{aligned} T_{i,j,k}^{m+1} =& T_{i,j,k}^{m} + 2\, F_{0z} H_z (T_{air} - T_{i,j,k}^{m}) + 8 F_{0z} R_z T_{air}^3 + \\ & 2 F_{0z} S_z A q_{sun} + 2 F_{0z} (T_{i,j,k+1}^{m} - T_{i,j,k}^{m}) + \\ & F_{0x} (T_{i-1,j,k}^{m} + T_{i+1,j,k}^{m} + T_{i,j-1,k}^{m} + T_{i,j+1,k}^{m} - 4 T_{i,j,k}^{m}) \end{aligned} \tag{2.73}$$

where we have assumed that $\Delta x = \Delta y \neq \Delta z$ and: come

$$R_z = \frac{\sigma \epsilon}{k} \Delta z \quad S_z = \frac{\Delta z}{k} \quad H_z = h S_z \tag{2.74}$$

Note that if the grid is uniform, i.e., $\Delta x = \Delta y = \Delta z_1 = \Delta z_2$, then Eqs. (2.71), (2.73) are converted in Eqs. (2.42), (2.36).

Non-uniform grids in FD-TD technique have been widely used for electromagnetic simulations, (Brankovic et al., 1992; Paul et al., 1994; Navarro et al., 1996; Dey et al., 1997; Falconer and Tripathi, 1997; Ala-Laurinaho et al., 1997; Kermani and Ramahi, 2006). The main reason for their use is that they reduce the computational load of the FD-TD technique allowing to increase the spatial resolution in regions where sharp changes of the variable being simulated are expected, (de Rivas, 1972; Navarro et al., 1996). The use of non-uniform grids also allows to save time and memory resources as the amount of point grids is reduced. Usually, when a PDE is solved using FD-TD with non-uniform grids, the local truncation error at the mesh points will be of first order in Δx and Δt. Some attempts have been made to reduce the magnitude of this error, for 1D systems, by using grids whose cells sizes vary slowly in the spatial domain. In some schemes the grid size change following this restriction:

$$\Delta x_i = \Delta x_{i-1} + \mathcal{O}(\Delta x_{i-1}^2) \tag{2.75}$$

where Δx_i is the size of the cell i. In (Sundqvist and Veronis, 1970) the function used to generate the grid was:

$$\Delta x_i = \Delta x_{i-1}(1 + \beta \frac{\Delta x_{i-1}}{L}) \tag{2.76}$$

that it is used to solve a 1D problem, where β is a constant and $0 < x < L$. The case with $\beta = 0$ corresponds to an uniform grid. Depending on the specific problem there

are many choices to choose the variation of the grid. However, obtaining a function to generate a grid for each specific problem can be a very long time-consuming process. In (Sundqvist and Veronis, 1970) a detailed study of the grid variation influence on the truncation error was made. The main conclusion of this work is that a grid scheme which fulfils:

$$\Delta x_i - \Delta x_{i-1} = \mathcal{O}(\Delta x_{i-1}^2) \tag{2.77}$$

warrants the numerical precision of the FD-TD algorithm. It is shown in (de Rivas, 1972) that varying the size of the grid intervals slowly and monotonically, that is, using the function $f = f(\phi)$, which is a smooth function of ϕ, where:

$$\Delta x_i = f(\phi_{i+1}) - f(\phi_i) \tag{2.78}$$

and

$$\Delta\phi = \phi_{i+1} - \phi_i = constant \tag{2.79}$$

gives second order approximation of the second derivative, since the truncation errors due to the non uniformity of the grid are of second order in $\Delta\phi$. Thus, a convenient choice is:

$$f = P_n(\phi) \tag{2.80}$$

where P_n is a polynomial.

Let us see how the use of non-uniform grids introduce an error on the computation of the PDE equation. For this discussion we will consider the 1D case:

$$T_{i+1}^m = T_i^m + \frac{\Delta x}{1!}\frac{\partial T_i}{\partial x} + \frac{(\Delta x)^2}{2!}\frac{\partial^2 T_i}{\partial x^2} + \frac{(\Delta x)^3}{3!}\frac{\partial^3 T_i}{\partial x^3} + \mathcal{O}(\Delta x^4) \tag{2.81}$$

$$T_{i-1}^m = T_i^m - \frac{\Delta x}{1!}\frac{\partial T_i}{\partial x} + \frac{(\Delta x)^2}{2!}\frac{\partial^2 T_i}{\partial x^2} - \frac{(\Delta x)^3}{3!}\frac{\partial^3 T_i}{\partial x^3} + \mathcal{O}(\Delta x^4) \tag{2.82}$$

If the spatial discretization step is constant then we can approximate the spatial second derivative, adding the previous equations, as:

$$\frac{\partial^2 T_i^m}{\partial x^2} = \frac{T_{i+1}^m - 2T_i^m + T_{i-1}^m}{(\Delta x)^2} + \mathcal{O}(\Delta x^2) \tag{2.83}$$

However, if the discretization steps are different, the term of order Δx in Eq. (2.81) it is not canceled with the term of order Δx of Eq. (2.82) and the following expression, assuming that Δx_1 and Δx_2 are the spatial discretization steps, is obtained:

$$\frac{\partial^2 T_i^m}{\partial x^2} = \frac{T_{i+1}^m - 2T_i^m + T_{i-1}^m}{\frac{\Delta x_1^2}{2} + \frac{\Delta x_2^2}{2}} + \frac{\Delta x_1 - \Delta x_2}{\frac{\Delta x_1^2}{2} + \frac{\Delta x_2^2}{2}}\frac{\partial T_i^m}{\partial x} + \mathcal{O}(\Delta x) \tag{2.84}$$

Thus, the non uniform grid produces an error in the approximation of the spatial second derivative of order 0, the term that multiplies to the first spatial derivative. Analyzing the expression we can see that if the grid variation is low then this introduced error is reduced. Moreover, if the variation of the grid follows Eq. (2.77), then we have again an approximation of the second spatial derivative of order Δx^2.

Now we will analyze the stability condition in this case. If we group the terms of Eq. (2.71) which multiply to $T^m_{i,j,k}$, we have the following stability condition:

$$1 - \alpha\Delta t \left(\frac{4}{(\Delta x)^2} + \frac{2}{\Delta z_1(\Delta z_1 + \Delta z_2)} + \frac{2}{\Delta z_2(\Delta z_1 + \Delta z_2)} \right) > 0 \tag{2.85}$$

If $\Delta d = min\,(\Delta z_1, \Delta z_2, \Delta x)$ then:

$$1 - \alpha\Delta t \left(\frac{4}{(\Delta x)^2} + \frac{2}{\Delta z_1(\Delta z_1 + \Delta z_2)} + \frac{2}{\Delta z_2(\Delta z_1 + \Delta z_2)} \right) > 1 - 6\frac{\alpha\Delta t}{(\Delta d)^2} \tag{2.86}$$

And thus the stability condition can be written as:

$$F_0 = \frac{\alpha\Delta t}{(\Delta d)^2} < \frac{1}{6} \tag{2.87}$$

Summarizing, the non-uniform grid is useful to reduce the computing time because the number of layers that must be computed is reduced or it can increase the spatial resolution in regions where sharp changes happen. However, it introduces an error in the equations that is minimized if an appropriated grid variation function is chosen. Another drawback of the non-uniform grid is that their use is very dependent on the considered problem and it is difficult to obtain a general expression of grid variation. Thus, a compromise between the precision and the number of layers to be computed must be found.

2.4.4 Non-uniform grid and media

The more general case that we can consider is the interface between two different media and where a non-uniform grid is used in the z direction. This situation is reflected in Fig. 2.4.4. If we consider an internal node, then the conservation energy law has this form:

$$\Delta x\Delta y \left(\frac{\rho_A c_A \Delta z_1}{2} + \frac{\rho_B c_B \Delta z_2}{2} \right) \frac{\partial T}{\partial t} = \sum_n q_n \tag{2.88}$$

The equations for the heat fluxes that come from the different nodes can be written as:

$$q_1 = k\frac{k_A\Delta z_1 4 + k_B\Delta z_2}{2}\Delta y\frac{T^m_{i+1,j,k} - T^m_{i,j,k}}{\Delta x} \quad q_2 = k\frac{\Delta z_1 + \Delta z_2}{2}\Delta y\frac{T^m_{i-1,j,k} - T^m_{i,j,k}}{\Delta x} \tag{2.89}$$

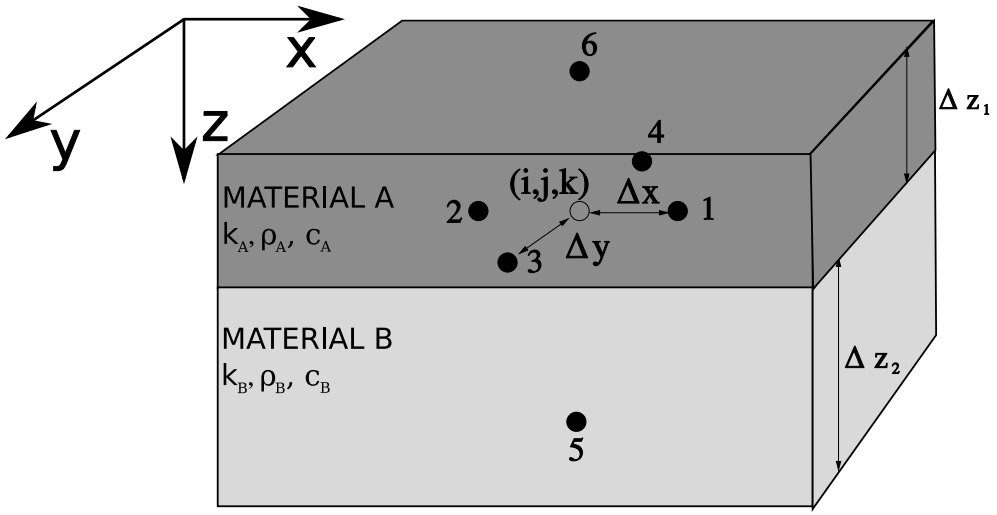


Figure 2.15: Scheme of a discretization with a non-uniform grid in the z direction and an interface between two different media.

$$q_3 = k\frac{\Delta z_1 + \Delta z_2}{2}\Delta x\frac{T^m_{i,j+1,k} - T^m_{i,j,k}}{\Delta y} \quad q_4 = k\frac{\Delta z_1 + \Delta z_2}{2}\Delta x\frac{T^m_{i,j-1,k} - T^m_{i,j,k}}{\Delta y} \tag{2.90}$$

$$q_5 = k\Delta x\Delta y\frac{T^m_{i,j,k+1} - T^m_{i,j,k}}{\Delta z_1} \quad q_6 = k\Delta x\Delta y\frac{T^m_{i,j,k-1} - T^m_{i,j,k}}{\Delta z_2} \tag{2.91}$$

Thus, introducing these expressions in Eq. (2.88), we obtain the equation to update an internal node temperature:

$$\begin{aligned} T^{m+1}_{i,j,k} =& T^m_{i,j,k} + F_{oxy}(T^m_{i+1,j,k} + T^m_{i-1,j,k} + T^m_{i,j+1,k} + T^m_{i,j-1,k} - 4T^m_{i,j,k}) \\ & F_{0z1}(T^m_{i,j,k-1} - T^m_{i,j,k}) + F_{0z2}(T^m_{i,j,k+1} - T^m_{i,j,k}) \end{aligned} \tag{2.92}$$

where:

$$F_{0xy} = \frac{\Delta t}{(\Delta x)^2}\frac{k_A\Delta z_1 + k_B\Delta z_2}{\frac{k_A\Delta z_1}{\alpha_A} + \frac{k_B\Delta z_2}{\alpha_B}} F_{0z1} = \frac{2\Delta t}{\Delta z_1}\frac{k_A}{\frac{k_A\Delta z_1}{\alpha_A} + \frac{k_B\Delta z_2}{\alpha_B}} F_{0z2} = \frac{2\Delta t}{\Delta z_2}\frac{k_B}{\frac{k_A\Delta z_1}{\alpha_A} + \frac{k_B\Delta z_2}{\alpha_B}} \tag{2.93}$$

The expression for a surface node is given by Eq. (2.73). It can be noted that as a more general model is used a more complex set of equation is obtained. Following a similar reasoning than in the two previous section, in this case, the stability condition can be written as:

$$F_0 = \frac{\Delta t}{(\Delta d)^2}\frac{k_M\alpha_m}{k_m} < \frac{1}{6} \tag{2.94}$$

where $\Delta d = min\,(\Delta z_1, \Delta z_2, \Delta x)$, $k_M = max(k_A, k_B)$ and $\left(\frac{k_m}{\alpha_m}\right) = min\left(\frac{k_A}{\alpha_A}, \frac{k_B}{\alpha_B}\right)$.

2.5 Software-based FD-TD solvers

The use of the thermal model for landmine detection is a high consuming process for nowadays computers. To overcome this limitation, some authors have developed different techniques with the aim of reducing the computing time. Here we will show two different approaches: reduction of size and duration of the experiments and efficient numerical methods for solving the thermal model.

2.5.1 Reduction of size and duration of experiments

In (Muscio and Corticelli, 2004) the authors presented a method by which the heating and cooling cycles of a soil with a buried mine can be replicated with reduction of size and time. The final objective of this work is to reproduce indoors, quickly and effortlessly, the outdoor conditions of any place where the IRT applied to the detection of buried land mines must be performed.

The authors argue that the main difficulty of of thermographic experiments is that the period of the thermal cycle is very long, 24 hours, inadequate to obtain a complete set of measurements. Due to the wide variability of weather conditions it is complicated to perform an on-field campaign in which the various parameters of the thermal model are adjusted selectively. As a result, it is impossible to carry out extensive sensitivity analyzes by outdoor experiments. On the other hand, indoor test would be too bulky and time-consuming to be manageable.

The authors have considered the cylindrical symmetry of the buried landmine, see Fig. 2.16. Thus, the heat equation in cylindrical coordinates is as follows:

$$\frac{\partial T(r,z,t)}{\partial t} = \alpha_{Soil}\left[\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) + \frac{\partial^2 T}{\partial z^2}\right] \tag{2.95}$$

$$\frac{\partial T(r,z,t)}{\partial t} = \alpha_{Mine}\left[\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) + \frac{\partial^2 T}{\partial z^2}\right] \tag{2.96}$$

$$\tag{2.97}$$

where r and z are the cylindrical coordinates. To perform the size and time reduction of the experiments the authors have performed a dimensional analysis of the equations and boundary conditions that govern the thermal behavior of the soil. The *Buckingham* π theorem is a key theorem in dimensional analysis, (Dee, 1998; Potter and Wiggert, 2002).

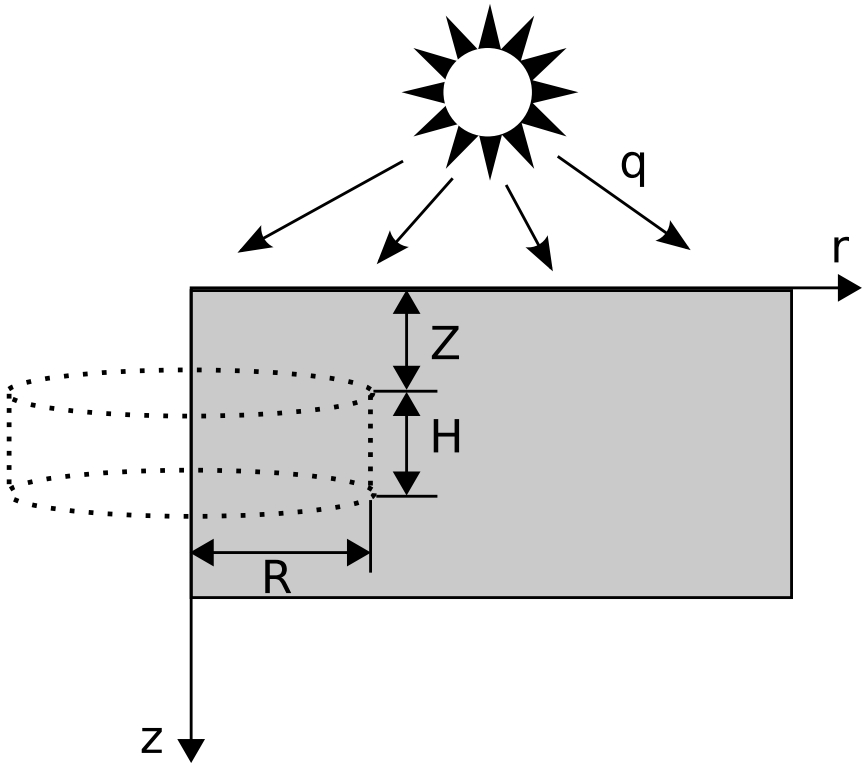


Figure 2.16: Cylindrical symmetry of the buried landmine.

This theorem states that if a dependent variable of a physical problem (temperature) can be expressed as a function of $n-1$ of independent physical variables, and m is the number of basic dimensions included in those variables, $(n-m)$ dimensionless groups of variables, called π-terms, can be identified and related by a relationship such as:

$$g(\pi_1, \pi_2, ...\pi_{n-m}) = 0 \tag{2.98}$$

where the dependency is included in π_1 and the remaining π-terms contain only independent variables. From this equation it is possible to state that experiments with same values of dimensionless groups have same dimensionless results. Thus, a reduced scale reproduction of an on-field test can be performed in the laboratory. The two experiments must be governed by the same dimensionless relationship and the dimensionless parameters must have equal values.

The non-dimensional heat equation for the soil and the mine are now:

$$\frac{\partial T^*(r^*, z^*, t^*)}{\partial t^*} = \frac{\alpha_{Soil} t_0}{R^2} \left[\frac{1}{r^*} \frac{\partial}{\partial r^*} \left(r^* \frac{\partial T^*}{\partial r^*} \right) + \frac{\partial^2 T^*}{\partial z^{*2}} \right] \tag{2.99}$$

$$\frac{\partial T^*(r^*, z^*, t^*)}{\partial t^*} = \frac{\alpha_{Mine}}{\alpha_{Soil}} \frac{\alpha_{Soil} t_0}{R^2} \left[\frac{1}{r^*} \frac{\partial}{\partial r^*} \left(r^* \frac{\partial T^*}{\partial r^*} \right) + \frac{\partial^2 T^*}{\partial z^{*2}} \right] \tag{2.100}$$

$$\tag{2.101}$$

where $T^* = (T - T_0)/\Delta T_0$, $t^* = t/t_0$, $r^* = r/R$ and $z^* = z/R$. The mine radius, R, is the reference value for the geometrical non-dimensionalization. The temperatures are related to an arbitrary reference temperature T_0 and the time to an arbitrary time reference t_0. These equations, together with the boundary conditions provide a set of parameters that allows to control the size and duration of the experiments. For example, reducing the cycle period from 24 hours to 3 hours requires shrinking the mine radius about one third of its actual size. The authors have performed several indoor reduced experiments and have achieved similar results to the on-field measurements.

2.5.2 Efficient numerical methods

In (Thanh et al., 2006) the authors propose an efficient numerical method to solve the thermal model applying finite difference methods. They have considered the advantages and disadvantages of the two finite difference schemes: explicit and implicit. Explicit finite difference scheme is very simple however it requires small time step size to guarantee the stability of the solution. On the other hand, the implicit scheme is stable for any time step size but it requires to solve large matricial systems, which is a very-high time-consuming task. Thus, the authors have chosen a splitting method, which avoids the disadvantages of the explicit and implicit schemes The idea of splitting schemes is to replace a complicated problem by a group of simpler problems, (Marchuk, 1975; Samarskii, 2001). This technique allows to obtain the advantages of being stable for an arbitrary time step size and it is faster than explicit and implicit schemes.

In this technique the volume of soil Ω is divided into parallelepipeds by the planes $x_i = m_i h_i$, with $m_i = 0, 1, ...N_i$; $h_i = l_i/N_i$, where l_i is the length of the edge of ω in the x_i direction, $i = 1, 2, 3$. Applying this technique to the thermal model they obtain a matricial system of equations which is absolutely stable and converges to the solution with the first order in x and 17th order in t.

2.6 Summary

In this chapter we have introduced the FD-TD method to solve numerically PDE equation. The use of this method presents limitations due to its high requirements in terms of memory and power computation. To overcome these issues several hardware implementations were proposed, most of them in the field of electromagnetic simulations. These proposals speed up the computations compared to software solutions. We have done a review of these implementations showing the main contributions of them. After this, we have shown obtained algebraic equations once the FD-TD method is applied to the thermal model for several situations. Finally we have introduced some other works in the field of IRT that tries to reduce the computing. The first one, (Muscio and Corticelli, 2004), propose a time and size reduction of the experiments. The duration of experiments, in our case, it is not a problem as the detection process has reduced the simulation time to 1 hour in the sunset, (López, 2003; López et al., 2004). The second technique follows the same direction of the first one, reduce the computing time of complete diurnal cycles, partitioning the full problem into small subsets to speed up the computations. However the use this mixed explicit-implicit scheme need that a relative large number of equations must be solved in parallel. As we are looking for a hardware implementation an explicit scheme is preferred.

As been said in the previous chapter, a software solution for mine detection processing time using IRT is not useful for on-field applications. Thus, we propose a hardware implementation into an FPGA which is discussed in detail in the following chapter.

Chapter 3

Heat equation hardware solver

3.1 Introduction

Modeling of physical phenomena often involves the use of complex sets of equations whose computational solution has demanding requirements in terms of memory and computing power. The FD-TD method provides a powerful technique that has been used widely in electromagnetic simulations, (Yee, 1966), in a broad range of applications, such as antennas design, (Stutzman and Thiele, 1997), ground penetrating radar, (Kosmas et al., 2002), studying the effects of electromagnetic radiation on alive tissues, (Ji et al., 2006), or in the analysis of electromagnetic effects on circuits, (Taflove, 1995). This technique also has applications on other fields, such as thermal modeling of objects for the detection of anomalies using NDE, (López et al., 2004). However, the use of this technique was limited until the past decade due to the high requirements in terms of memory and computing power; even for nowadays personal computers (PCs) it is an expensive technique, in terms of resources consumption. To avoid the limitations imposed by the PCs some hardware implementations of the FD-TD model were synthesized in hardware.

We include here a brief summary of the FD-TD hardware implementations, which were discussed in detail in the previous chapter. The first proposal of a FD-TD hardware implementation was reported in (Marek et al., 1992); in this work the authors presented a simulated hardware description language design acting as a co-processor or accelerator. In (Schneider et al., 2002a) the authors performed an FPGA implementation of the FD-TD method where the operations are performed with integer arithmetic. In this implementa-

tion the operations are performed in pipelined bit-serial arithmetic; this allows to make a mapping between node and processing element. However, bit-serial implementations have the drawback that operations are slower than using full bit arithmetic. In (Placidi et al., 2002) the authors carried out a standard cell digital implementation; they achieved an important speedup factor, nevertheless the simulations are restricted a small number of nodes. In (Durbano et al., 2003a) the first 3-D hardware implementation of a FD-TD solver is described; however the hardware solution was 9 times slower than the software one. The slowness of their implementation is due to the fact that they use floating-point arithmetics. The authors improved the hardware implementation, (Curt et al., 2007), achieving a speedup factor of two orders of magnitude in the best case, i.e., for big size computation when the data do not fit on the PC RAM and then it must be used the hard disk to store data, which slows down the computations. This implementation can compute up to 260 million nodes meshes as their hardware has 16 GB of RAM. In (Chen et al., 2006) another FD-TD hardware implementation is reported, where a speedup factor of 28 is obtained compared to a software solution, as integer arithmetic is used. In (EMPhotonics, 2007) a different solution for this problem is reported; they use a PC graphic board to perform the calculations taking advantage of nowadays boards memory and computing power. They developed a set of libraries to program the graphic board as an FD-TD accelerator for electromagnetic calculations.

Our goal is to reduce the computing time by performing a hardware implementation of a system capable of simulate the thermal model. In this Chapter, a description of the hardware onto which the thermal model has been projected is made. After this, a detailed description of the system architecture and its implementation that have been developed is made. The different steps of the system development have been published in (Pardo et al., 2006, 2007, 2008b,a,c).

3.2 Hardware Description

We have considered two digital platforms to project the system onto: FPGA and Digital Signal Processor (DSP). Both are digital reconfigurable platforms, but the structure of the two platforms is quite different. A brief introduction to the architecture of these two platforms is made showing the differences between them.

Let us first introduce FPGA. There are a lot of FPGA vendors, the most important are

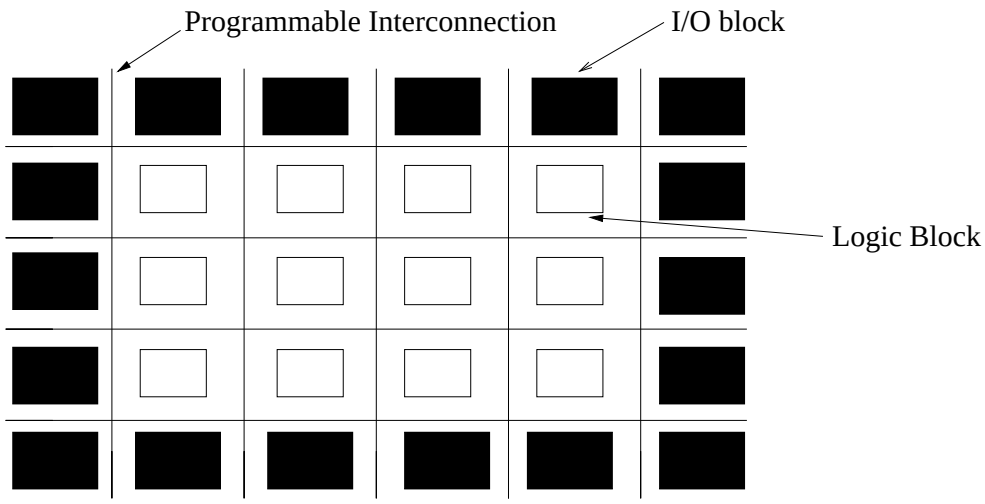


Figure 3.1: Standard FPGA structure.

Xilinx, Altera, Atmel and AT&T. In spite of the existing differences between all vendor models, most FPGAs are organized as an array of configurable logic element, programmable interconnections between logic elements, I/O pins and other resources, see Fig. 3.1. The logic elements can be programmed to give the desired functionality. Usually more than one configurable logic element will be part of our design, thus the connection between the logic elements is made through the programmable interconnections. Finally, the I/O pins allow the FPGA to establish a connexion with the outside world.

The DSP is another digital platform and the most important DSP vendors are Texas Instruments, Freescal and Agere Systems. As happens with FPGA, each DSP vendor has its own architecture, but there are some aspects that are common to all of them: specialized high speed arithmetic, data path communication with the outside world and memory architectures (how the access to DSP memory is made). The high speed arithmetic is optimized to perform additions and multiplications in floating or integer arithmetic, depending on the DSP model, and the data path communication allows the connection of the DSP to the outside world.

The design flow for and FPGA is closer to the hardware design, where the programmer must take into account the hardware with which he or she is dealing. On the other hand, DSP programming is closer to software and the programmer does not have to deal with hardware issues, at least not at as low a level as with the FPGA. FPGAs are more versatile because their functionality can be totally defined by the user, while DSPs have specialized units that perform some operations very quickly. Another difference between these two

Figure 3.2: Image of the RC2000 card.

platforms is that with an FPGA a higher grade of parallelism than with a DSP can be achieved; with a DSP we have the restriction of executing one or two instructions per clock cycle. One important issue, mainly in portable devices, is power consumption, which is directly proportional to the clock frequency at which the system operates. So the ability of the FPGA to split a task in various parallel tasks, reducing the clock rate, becomes an important factor in power consumption. However, FPGAs are significantly more expensive than DSPs and the design cycle for an FPGA is longer than for a DSP, but nowadays the price of the FPGA is getting down and there are new FPGA programming languages, such as Handel-C, (Celoxica, 2005a) or System-C, (Initiative, 2003), that reduce the design flow of an FPGA design.

Taking into account the characteristics of these two hardware platforms and the nature of our system an FPGA hardware platform was chosen, which will provide a high grade of parallelism to the FD-TD accelerator.

The hardware used to project the thermal model onto is an RC2000 card from Celoxica, see Fig. 3.2; which is made up of a RC2000 PMC card and a PCI-PMC carrier card, which allows the connection of the card to the PCI bus of a personal computer (HOST). The RC2000 PMC card provides the following hardware:

- Xilinx Virtex-II xc2v6000-4 FPGA.

- Six banks of Zero Bus Turnaround (ZBT) RAM of 2 MB each one, accessible by both FPGA and HOST.
- Two banks of ZBT RAM 4MB each one, accessible only to the FPGA.
- 16 MB Flash for configuration storage.
- Two programmable clocks.

Six ZBT memory banks are accessible from the FPGA and from the HOST, via the PCI bus, while the other two memory banks are only accessible by the FPGA. The Flash memory allows to store multiple FPGA configuration and use them to program the FPGA. One of the clocks can be set at a clock rate between 40 kHz and 100 MHz, while the other can be set to a clock frequency between 25 MHz and 66 MHz. A brief description of the ZBT memory banks and of FPGA architecture is made in the following sections. A diagram of the different components of th RC2000 card is shown in Fig. 3.3.

3.2.1 Virtex-II FPGA

The Virtex-II family from Xilinx is an FPGA platform developed for high performance applications from low-density to high-density designs based on IP cores and customized modules. The Virtex-II has various configurable elements; as can be seen on Fig. 3.4, its architecture is made up of Input/Output Blocks (IOBs), internal configurable logic and routing resources. The programmable IOBs provide the interface between package pins and the internal configurable logic. The configurable logic allows to configure the FPGA with the desired functionality. The internal configurable logic includes four elements organized in a regular array:

- Configurable Logic Blocks (CLBs): provide functional elements for combinatorial and synchronous logic, including basic storage elements.
- Block select-RAM memory modules: provide large 18 Kbit storage elements of dual-port RAM.
- Multiplier blocks: 18-bit x 18-bit dedicated multipliers.
- Digital Clock Manager (DCM): provides self-calibrating, fully digital solutions for clock distribution delay compensation, clock multiplication and division, coarse and fine-grained clock phase shifting.

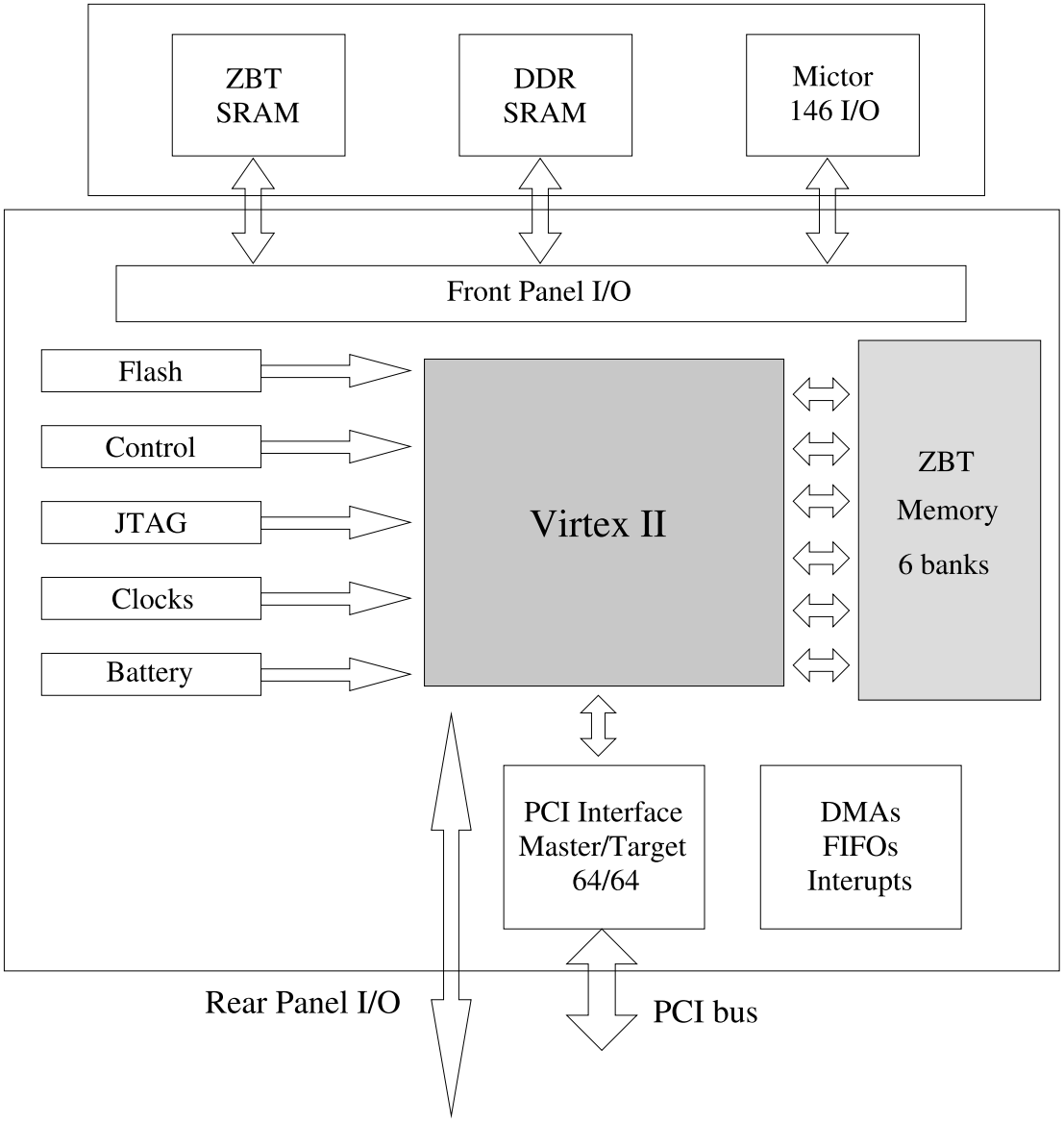


Figure 3.3: RC2000 block diagram (Reprinted from RC2000 user manual).

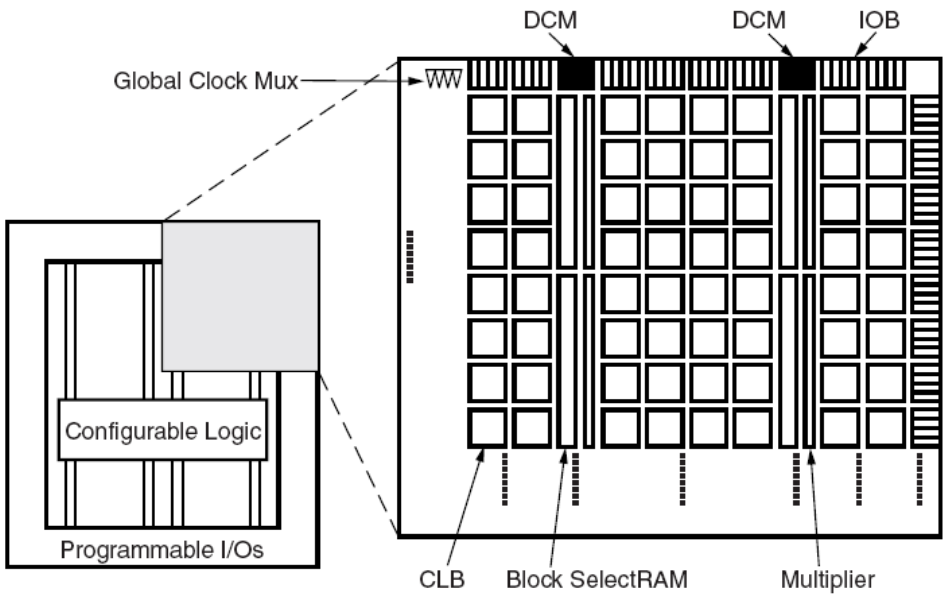


Figure 3.4: Virtex-II architecture (Reprinted from Virtex-II datasheet).

- Routing resources: provide connection capabilities between configurable logic blocks.

In this section we will show the basic aspects components of the Virtex-II and its architecture; for a more detailed description see (Xilinx, 2004).

IOBs

The IOBs allow communication of the CLBs with the outside world. Each IOB block includes six storage elements, see Fig. 3.5, and each one can be configured either as an edge-triggered D-type flip-flop or as a level-sensitive latch. The configuration of each storage elements is independent from the configuration of the other storage elements. Virtex-II IOBs are provided in groups of two or four on the perimeter of each CLB, see Fig.3.4.

CLBs

The Virtex-II CLBs are organized in a regular array and they are used to build combinational and synchronous logic designs. A CLB element comprises 4 similar slices, see Fig. 3.6, with fast local feedback within the CLB. Each slice includes two 4-input function generators, carry logic, arithmetic logic gates, wide function multiplexers and two storage

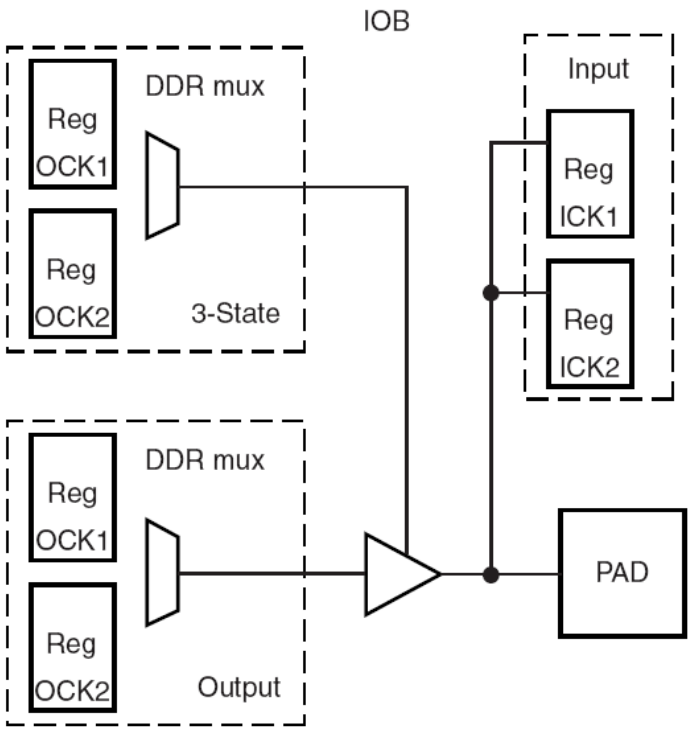


Figure 3.5: Virtex-II IOB block (Reprinted from Virtex-II datasheet).

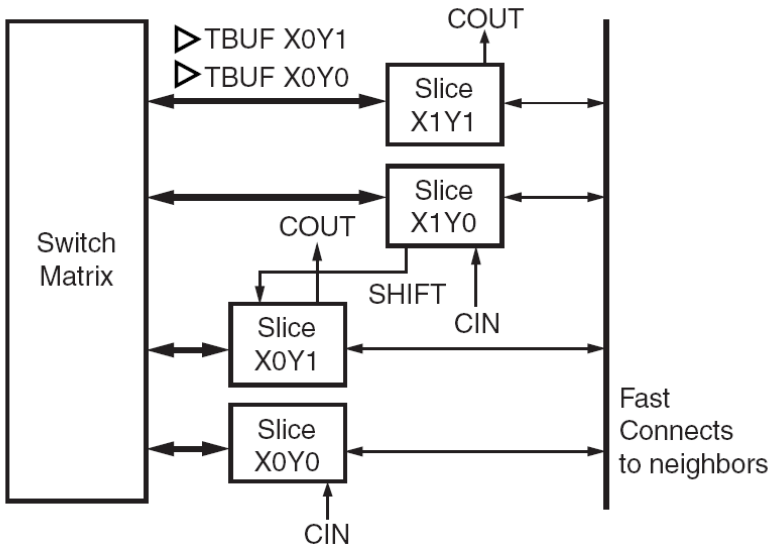


Figure 3.6: Virtex-II CLB element (Reprinted from Virtex-II datasheet).

elements. In Fig. 3.7 it is shown the half part of a Virtex-II slice where we can distinguish the different parts of the CLB.

As shown in Fig. 3.8, each 4-input function generator is programmable as a 4-input LUT, 16 bits of distributed select-RAM memory, or a 16-bit variable-tap shift register element. Thus, a CLB operating as a LUT table can implement any 4-input function where the propagation delay is independent of the implemented function. Additionally, each slice contains logic, MUXFx and MUXF5 that combine function generator to provide any function of five, six, seven or eight inputs, see Fig. 3.8. The storage elements in each slice can be configured as edge-triggered D-flip-flops or as level sensitive latches. Each function generator (LUT) of the CLB can implement a 16 x 1-bit synchronous RAM resource called a distributed select-RAM element. Each function generator can also be configured as a 16-bit shift register. Longer register can be built by connecting in cascade the outputs of the LUTs. Each CLB element is tied to a switch matrix to access the general routing matrix, as shown in Fig. 3.6.

Select-RAM blocks

Virtex-II devices incorporate large amounts of 18 Kbit block select-RAM. These complement the distributed select-RAM resources that provide shallow RAM structures implemented in CLBs. Each Virtex-II block select-RAM is an 18 Kbit true dual-port RAM with two independently clocked and independently controlled synchronous ports that access a common storage area.

18-bit x 18-bit multipliers

Virtex-II FPGAs incorporate many embedded 18-bit by 18-bit 2´complement multiplier blocks. These multipliers are optimized for high-speed operations and have a lower power consumption compared to 18-bit x 18-bit multipliers built with CLBs.

Digital Clock Manager

The Virtex-II DCM offers the following management features:

- Clock De-skew: generates new system clocks (either internally or externally to the

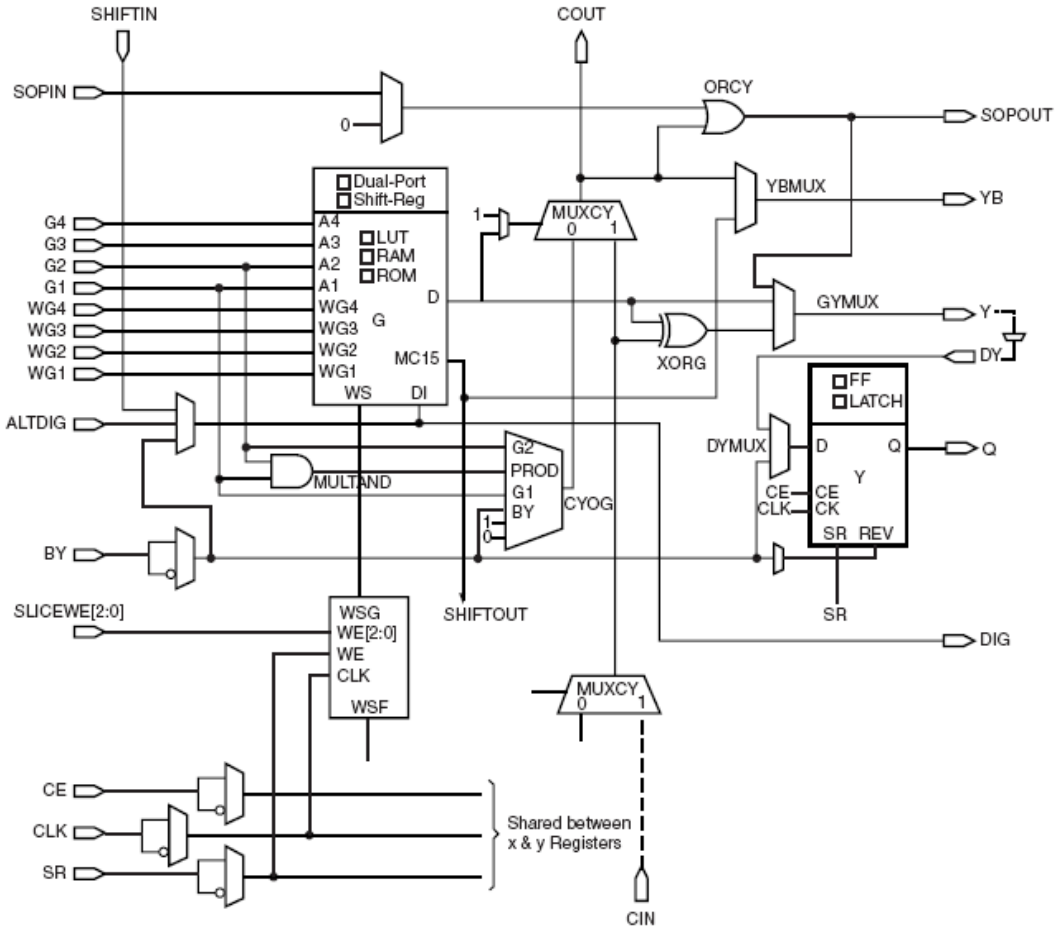


Figure 3.7: Half part of a Virtex-II slice (Reprinted from Virtex-II datasheet).

FPGA), which are phase-aligned to the input clock, thus eliminating clock distribution delays.

- Frequency Synthesis: generates a wide range of output clock frequencies, performing very flexible clock multiplication and division.
- Phase Shifting: provides both coarse phase shifting and fine-grained phase shifting with dynamic phase shift control..

The DCM utilizes fully digital delay lines allowing robust high-precision control of clock phase and frequency. It also utilizes fully digital feedback systems, operating dynamically to compensate for temperature and voltage variations during operation.

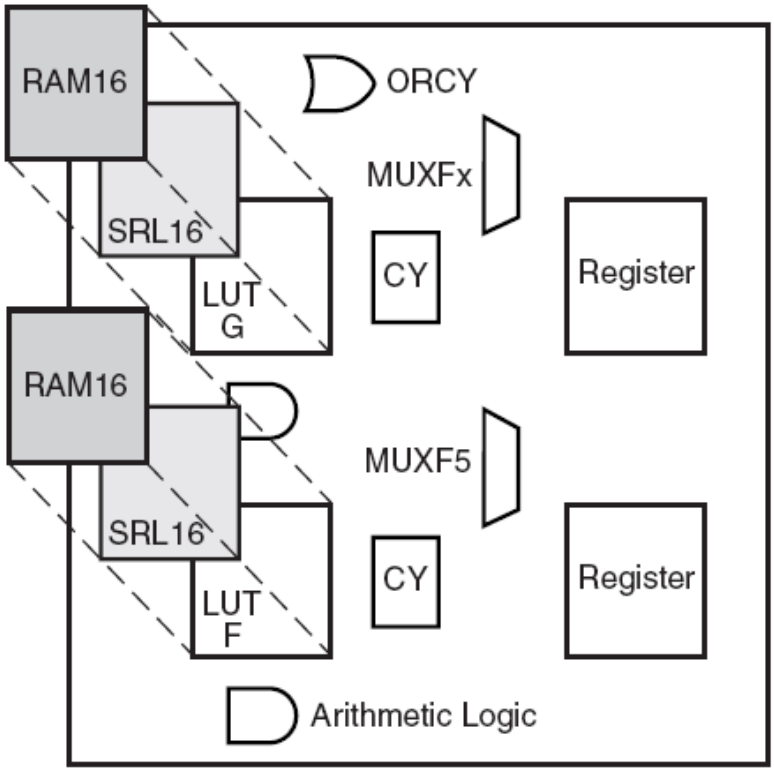


Figure 3.8: Virtex-II slice configuration (Reprinted from Virtex-II datasheet).

Routing resources

The routing resources allow to connect the CLBs required to build the desired functionality. The Virtex-II has different types of connections:

- 24 Horizontal and Vertical Long lines. These routing lines are bidirectional wires that distribute signals across the FPGA.
- 120 horizontal and Vertical Hex lines. These lines route signals to every third or sixth block in all directions.
- 40 horizontal and vertical double lines. Theses lines route signals to every first or second block in all directions.
- 16 direct connections. These lines route signal to all neighboring blocks.

Table 3.1: Different Virtex-II FPGA platforms and their main features.

Device	Gates	CLBs	Dist. RAM(Kbits)	Multiplier Blocks	Sel RAM(Kbits)	DCMs	I/O Pads
XC2V40	40K	8 x 8	8	4	72	4	88
XC2V80	80K	16 x 8	16	8	144	4	120
XC2V250	250K	24 x 16	48	24	432	8	200
XC2V500	500K	32 x 24	96	32	576	8	264
XC2V1000	1M	40 x 32	160	40	720	8	432
XC2V1500	1.5M	48 x 40	240	48	864	8	528
XC2V2000	2M	56 x 48	336	56	1008	8	624
XC2V3000	3M	64 x 56	448	96	1728	12	720
XC2V4000	4M	80 x 72	720	120	2160	12	912
XC2V6000	**6M**	**96 x 88**	**1,056**	**144**	**2592**	**12**	**1104**
XC2V8000	8M	112 x 104	1,456	168	3024	12	1108

- 8 Fast Connects. These are the internal CLB local interconnections from LUT outputs to LUT inputs.

On Table 3.1 a summary of the main characteristic of Virtex-II FPGA series can be seen. The FPGA mounted in the RC2000 is emphasized in bold. The RC2000 is offered with various FPGA: Xilinx 2V3000, Xilinx 2V6000 and Xilinx 2V8000. It is obvious that one factor to be taken into account is the price of the card, which grows up as the size of the FPGA is increased. Thus, a compromise between size and price must be done. The Xilinx 2V6000 fills this compromise, with a reasonable price and characteristics that allows to implement complex problems. One of the main reasons to choose this instead of the Xilinx 2V3000 is the amount of select-RAM which offers a bigger flexibility. The main advantages of the RC2000 card from Celoxica are its large amount of memory integrated in the card, compared to other development boards from AVNET, DALANCO or HITECH, and the use of the Handel-C language from Celoxica, which facilitates card programming.

3.2.2 ZBT Memory Banks

Due to the increasing functionality required of nowadays main memory subsystems the time that it takes a DRAM to switch between a read and write cycle or between a write and read cycle is becoming a critical factor. This time is commonly referred to a bus

turnaround time; delays in turning the bus around can result in costly dead bus cycles and reduce performance. In order to minimize dead bus cycles, fast (preferably zero) read-to-write or write-to-read bus turnaround time is required. The ZBT memory banks have been designed to eliminate dead bus cycles when turning the bus around between reads and writes or writes and reads. A ZBT bank is made up of an RAM array and the necessary control circuitry that ensures the elimination of the dead bus cycles. Address and control signals are applied to the ZBT during one clock cycle, and two clock cycles later the associated data cycle occurs, be it read or write. ZBT memory banks guarantees that a memory address can be read or written in two clock cycles. The structure of a ZBT memory bank from Integrated Device Technology can be seen on Fig. 3.9; where the RAM array and control circuitry that eliminates the dead cycles can be seen. The RC2000 can be bought with different configurations of memory banks, we chose the configuration which have the highest number of memory banks taking into account the nature of the problem which we are dealing.

3.3 FD-TD hardware solver

The system that simulates the behavior of the thermal model is made up of a software and a hardware part, see Fig. 3.10. The software part was developed in C++ and runs in a personal computer, while for the hardware part Handel-C, (Celoxica, 2005a), and VHDL, (VHDL, 1987) were used. A mixed solution VHDL/Handel-C solution was considered, where VHDL was used to describe the processing core and Handel-C to control the peripheral resources and communications with the software. Such a system was preferred because using classical hardware description languages, such as VHDL, higher performance is achieved compared to high level hardware description languages, (Sullivan and Saini, 2003; Ortigosa et al., 2006), while Handel-C makes I/O implementation significantly easier.

The thermal model has been projected onto an RC2000 card from Celoxica, described in the previous section, which has a Virtex-II xc2v6000-4 FPGA, 8 ZBT (Zero Bus Turn around) memory banks, labeled as *MBank k* (k=0, ... 7) in Fig. 3.10, and a PCI connector which allows the card to be attached to the PCI bus of a Personal Computer. In this chapter a detailed description of each part of the system architecture is made.

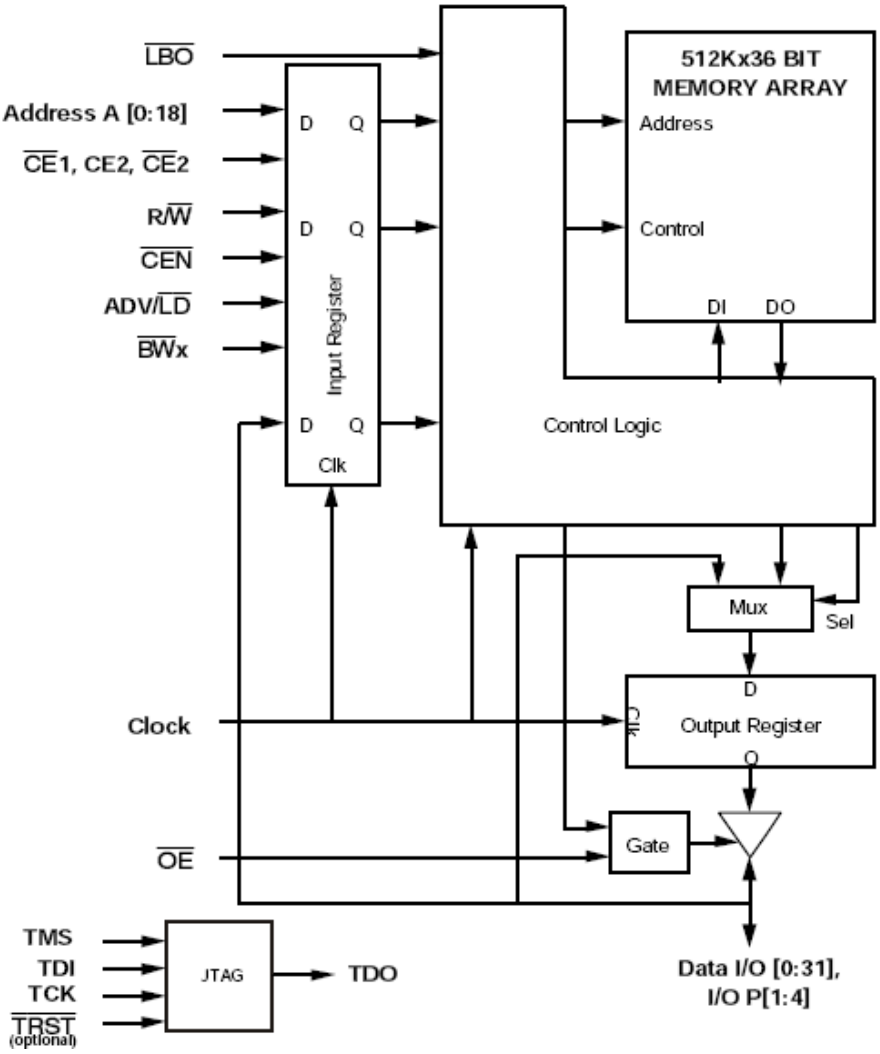


Figure 3.9: Structure of a ZBT RAM from IDT (Reprinted from IDT ZBT RAM datasheet).

3.3.1 HOST

In this architecture the HOST (Personal Computer) uploads to the memory banks of the RC2000 and to the select-RAM of the FPGA the data to be processed, i.e.: the initial temperatures of all the points of the volume, the thermal parameters of the soil, the boundary conditions (temperature of the air and solar radiation) and the parameters of the simulation, such as number of points of the volume on each direction, number of samples of the boundary conditions and total number of iterations. Using this scheme the necessity of making a Place&Route each time a new simulation is done is avoided because only new parameters must be loaded in order to perform a new simulation.

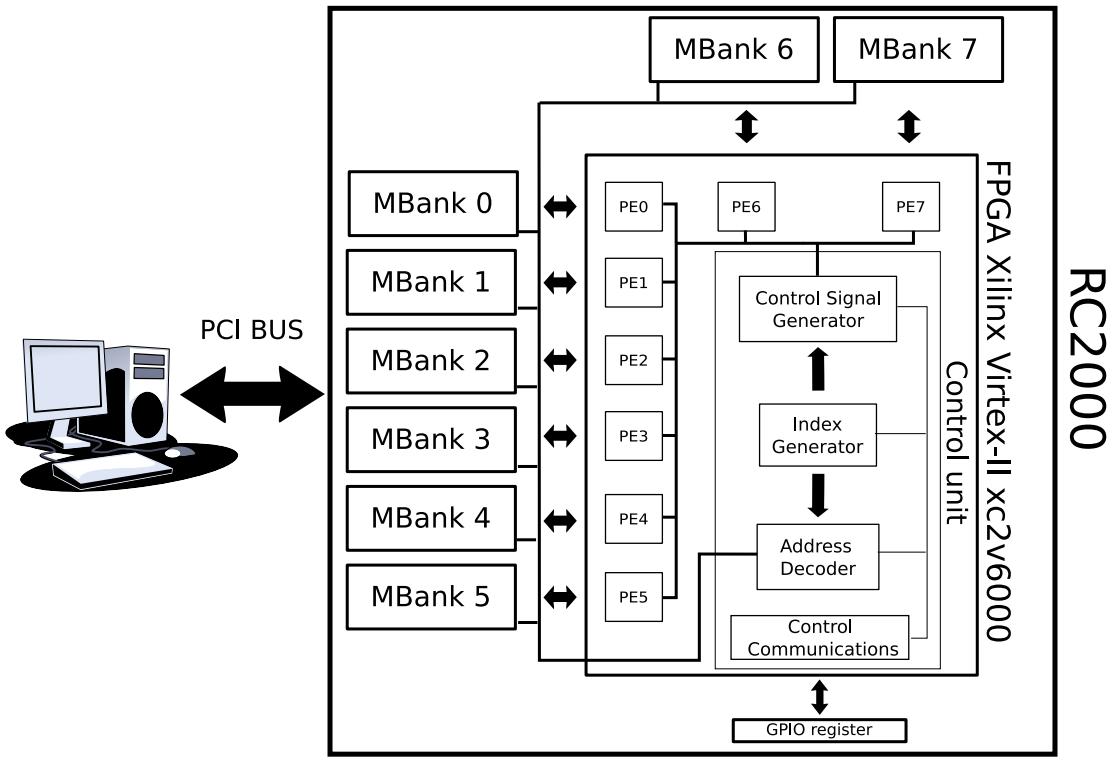


Figure 3.10: General view of the system and connections between all components.

When a thermal simulation of the soil for a long period of time is carried out the HOST must read the memory banks periodically in order to get intermediate results. The data that need to be collected correspond to the surface layer, as the IR sensor gives us information about this layer. These readings must be done in synchronization with the FPGA to avoid memory bank utilization conflicts. To this aim a synchronization protocol has been developed. Each time the HOST needs to read the memory banks, the FPGA stops the processing, releases the memory banks and sends a signal to the HOST indicating that the memory banks have been released. Once the HOST has read the data, it indicates to the FPGA that it can continue with the data processing. This synchronization is performed using the GPIO (General Purpose Input Output) register (32 bits wide) of the RC2000 card which allows the communication between the FPGA and the HOST, (Celoxica, 2005b). The synchronization is carried out writing and reading bits that indicate whether the memory banks have been released or not; to this aim Handel-C functions are used, (Celoxica, 2005a). A diagram of the communications between the HOST and the FPGA is shown in Fig. 3.11. Two functions are used to control it; the first is $GetGPIO(Channel, Value)$ which allows to read the bit position of the GPIO

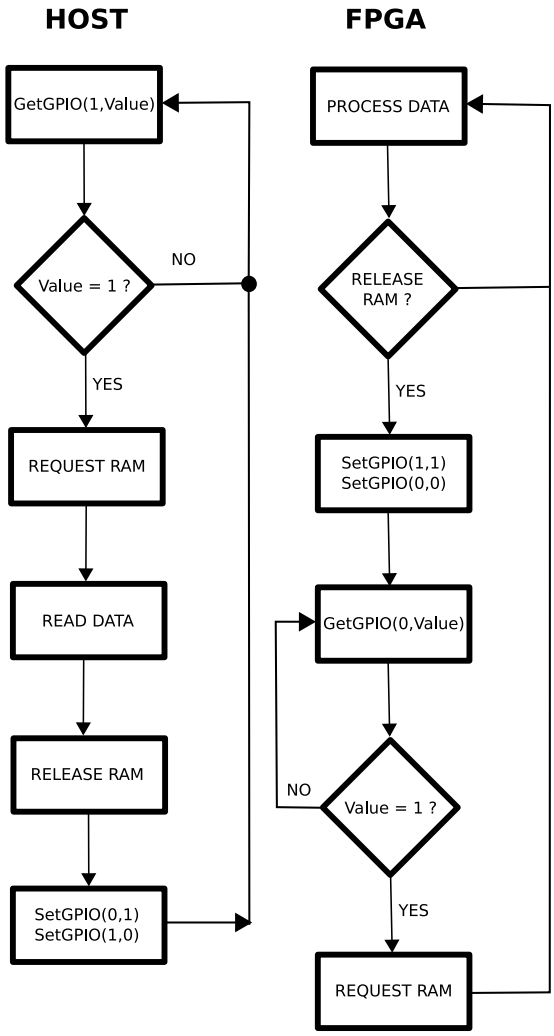


Figure 3.11: Handshaking protocol to share the use of the ZBT memory banks.

register given by $Channel$ and store it in the variable $Value$. The second function is $SetGPIO(Channel, Value)$ which writes in $Channel$ bit position of the GPIO the content of the variable $Value$. The number of iterations between two consecutive readings of the memory from the host is programmable. Once the FPGA has reached this number of iterations, it releases the memory banks, sets bit 1 of the GPIO and resets bit 0. In this moment the HOST reads bit 1 of the GPIO; the value 1, written by the FPGA, indicates that the memory banks have been released. Then, the HOST takes the ownership of the memory banks and reads the data. Once the reading has finished the HOST sets bit 0 of the GPIO and resets the bit 1. This indicates to the FPGA that the HOST has finished the reading and it takes the ownership of the memory banks. This process is repeated until

Table 3.2: SDK functions provided from Celoxica for the RC2000 card.

Function	Task
ADMXRC2ConfigureFromFile	Configure FPGA with a bit file
ADMXRC2MMapRequestBank	Request access tho the ZBT SRAM banks
ADMXRC2MMapReleaseBank	Release access to the ZBT SRAM banks
ADMXRC2MMapDoDMA	Initiates DMA HOST⇔ ZBT
ADMXRC2MMapGetGPIO	Read values from GPIO register
ADMXRC2MMAPSetGPIO	Write values to GPIO register

the desired number of iterations has been reached.

The HOST part was developed using C++ and the functions of the Platform Support Library (PSL) developed by Celoxica to program the RC2000 card. The functions of the library used to develop this part and a brief explanation of their behavior is given in Table 3.2. This is a flexible system because the user can change all the parameters of the simulation, avoiding the necessity of perform a Place&Route each time a parameter of the simulation is changed. As the system is fully reconfigurable, to perform a new simulation, with different parameters, the user only has to upload them to the FPGA.

Summarizing, the tasks of the HOST are three. In a first step the HOST uploads the data of the simulations to the ZBT memory banks; after, it reads the final or intermediate results of the simulations from the ZBT RAM banks and it programs the FPGA with the **.bit** file.

3.3.2 Hardware

The overall system includes the 8 memory banks of the RC2000 card and the Virtex-II xc2v6000-4 FPGA. The memory banks are used to store the temperatures of the nodes in the volume, while the rest of the data needed in the computations, i.e., the value of the measured temperature of the air and solar radiation, and the $F_0(i, j, k)$, R, S and H parameters in Eqs. (2.36)-(2.42) are stored in the FPGA select-RAM, (Xilinx, 2004). In this way more temperature nodes can be read in parallel, which increases the speedup. In this application we are considering two target categories for the identification of detected

Table 3.3: Handel-C functions provided from Celoxica to operate with the RC2000 card.

Function	Task
ADMXRC2MMapRequestBank	Request access to the ZBT banks
ADMXRC2MMapReleaseBank	Release access to the ZBT banks
ADMXRC2SSRAM*SetWriteAddress	Set the address of bank * to write to
ADMXRC2SSRAM*SetReadAddress	Set the address to read to
ADMXRC2SSRAM*ReadData	Read data from the ZBT bank
ADMXRC2SSRAM*WriteData	Write data to the ZBT bank
ADMXRC2MMapGetGPIO	Read the GPIO register
ADMXRC2MMAPSetGPIO	Write to the GPIO register

objects: soil and mine, thus with just one bit it is possible to code whether the node under considerations is soil or mine, which is reflected in different values of the α parameter, and therefore of F_0; if new categories should be considered more bits could be added to code them.

As was previously pointed, the system was designed using Handel-C and VHDL. VHDL was used to develop the processing core of the system, because a better performance and utilization of resources than that of Handel-C can be achieved, (Sullivan and Saini, 2003; Ortigosa et al., 2006). On the other hand, Handel-C was used to develop two tasks: the interface between the processing core and the memory banks and the synchronization between the HOST and the FPGA to read the memory banks. Handel-C provides functions to facilitate the access to the RC2000 memory banks and the communication with the HOST, (Celoxica, 2005a). This allowed us to make a faster development of the interfaces between the different components of the global system, FPGA-ZBT banks and FPGA-HOST, and to concentrate the efforts on the improvement of the processing core. The Handel-C functions used to developed the interfaces can be seen in Table 3.3.

The parameters of the simulation, uploaded by the HOST to the select-RAM of the FPGA, set the behavior of the VHDL processing core: number of points in each direction, number of samples of air temperature and solar radiation intensity, number of iterations between two consecutive samples of air temperature and solar radiation intensity and number of iterations between each HOST memory reading. Using this scheme we avoid the necessity of making a Place&Route each time a new simulation must be done. Thus,

Table 3.4: Processing time for a clock frequency of 45 MHz.

Amount of numbers to add	Time Small DMA (ms)	Time Large DMA (ms)
10000	3641	0.6
100000	34780	2.9
200000	71047	5.3
300000	107156	8.0
400000	141843	10.1
500000	192859	12.0

to perform a simulation with different parameters the HOST only has to upload the new parameters to the ZBT memory banks.

A test on the FPGA was performed to study which is the better way to transfer the intermediate results from the ZBT RAM banks to the HOST. To this aim a simple example was developed: a 32 bits adder was designed with VHDL and it was integrated into a system developed with Handel-C. In this example the data is read from a ZBT bank, sent to the VHDL adder module, and then the result is uploaded to another ZBT bank. Two different memory access schemes were employed by the HOST: in the first scheme each time the adder performs an addition the FPGA stops the processing until the HOST has read the result from memory and after the reading the FPGA continues processing the next addition. In the second scheme the FPGA performs the addition of all the values stored in memory and then the HOST reads the data written in the memory as one block. In the first case we are making small DMAs, while in the second case we are making one large DMA. On Table 3.4 we can see a comparison between these two memory access schemes. As can be seen in Table 3.4 the system in which small transfers of data are made increases the computing time several orders of magnitude compared to the other scheme. This is due to the fact that the establishment of the DMA transference is a time consuming process and therefore its continuous use slowness the processing of the data. Therefore it is preferable to perform large DMA data transference.

In the simulations we need to read intermediate results from the memory banks; thus, to read the data large DMAs transfers are preferred because the establishment of the DMA process is a very-high time-consuming process and once the channel to transfer the data has been established it is preferable to exploit it transferring more data.

3.4 FPGA implementation

In this section the architecture of the processing core is presented. As was said previously, the processing core has been developed with VHDL to obtain a high performance. In the design of the synthesized architecture it has to be taken into account that due to the high number of nodes involved in the computation (hundreds of thousands), a complete mapping volume node - processing element (PE) is obviously beyond the capability of any FPGA or even ASIC. Taking into account this issue, and considering the hardware in which the system is going to be synthesized (8 memory banks) we have developed an architecture with 8 PE. The memory banks store the temperatures that must be iteratively updated. A relationship PE/Memory bank is established, where each PE updates the temperatures of its corresponding memory bank.

We will now discuss all the aspects related to the FPGA implementations, and describe the most important functional blocks.

3.4.1 Arithmetic analysis and wordlength

In this section we will discuss the use of floating-point vs integer arithmetic in our system. In the literature there are implementations of the FD-TD algorithms for electromagnetics simulations, (Schneider et al., 2002a; Culley et al., 2005; Curt et al., 2007; Chen et al., 2006) or acoustic simulations, (Motuk et al., 2005). Some authors, (Culley et al., 2005; Curt et al., 2007), defend the use of floating-point arithmetic to avoid some problems, such as:

- The error quantization introduced with the fixed-point arithmetic.
- Due to the wide variation range of the parameters, in some cases the dynamic range affects to the precision of the computations.
- Nowadays new FPGA, such as Virtex-5, (Xilinx, 2006), incorporates DSP, which facilitates the use of floating-point arithmetic in FPGA designs.

However, other authors, (Schneider et al., 2002a; Roy and Banerjee, 2005; Chen et al., 2006), prefer the fixed-point arithmetic because an efficient and compact implementation is obtained and also the performance of the system is higher than using floating-point

arithmetic. Making a carefully quantization of the quantities involved in the computations a reduction in the quantization error can be achieved, (Roy and Banerjee, 2005). In addition, floating-point arithmetics introduces the overhead of normalizing the operands when addition and subtraction operations are performed. The difference in speed, in terms of million of operations per second (MOPS), can be of two orders of magnitude, (Lanuzza et al., 2005), between floating-point and fixed-points arithmetics. Taking into account these issues, and considering that the main objective of this work is to reduce the computing time, we chose fixed-point arithmetics. As will be seen, the achieved precision is enough to satisfy the requirements of the application.

All physical values measured from the infrared sensor and the probes are real numbers, making it necessary to transform them to a fixed-point representation. The longer the bitwidth used to represent the integer values, the higher the resolution, so the smaller the error. However, longer bitwidth data costs more hardware resources. Thus, we need to study the numerical results carefully to choose a suitable wordlength with relatively small error and hardware cost. A first step was to set the wordlength of the parameters involved in the computations. This task was performed using MATLAB, as a previous step to hardware implementation, restricting the operands to integers of different lengths. Each parameter of the algorithm is represented as an integer using the following transformation:

$$FIP = \left[2^{\beta} \cdot FPV\right] \tag{3.1}$$

where FPV is the floating-point value of the data, FIP is the fixed-point (integer) value of the data and β is a parameter used to perform the expansion to the integer value. The factor β is obtained taking into account the range of the data and the number of bits used to represent it. Thus, if we want to use M bits to represent a value, then the scaling factor is obtained as:

$$\beta = \left[log_2\left(\frac{2^M}{max\ FPV}\right)\right] \tag{3.2}$$

To perform the transformation from real to integer value it has to be taken into account that the node temperature range is $0 - 90$ ℃, and it is necessary to obtain, at least, a precision of the order of 0.1 ℃, to ensure the numerical stability of the solution and the precision of the computations. We must choose the number of bits used to represent all the data of the algorithm. Our study is focused on the representation of the temperature, that constitutes the large amount of data. The choice of the number of bits used to represented the temperature was done performing simulations and analyzing the differences between the simulations in double precision and those made with different integer wordlengths.

Floating-point to fixed-point quantization creates errors; these errors are caused by two main factors: arithmetic error from floating to fixed-point transformation and additional error created by the calculation from pieces of data which originally have error. Since we can not avoid the second type of error, except by changing the algorithm structure, we need to consider the two types of error as a whole. The relative error method compares all the floating-point data results and the corresponding fixed-point data results for the same model space, (Chen et al., 2004). The relative and absolute errors are calculated according to the following equations:

$$AE = |FIP - FLP| \tag{3.3}$$

$$RE = \frac{|FIP - FLP|}{|FLP|} \tag{3.4}$$

where AE is the absolute error and RE is the relative error. The average relative error reflects the difference of the fixed-point data from the original floating-point data. We have chosen the absolute error as an indicator of the correctness of the simulation because we are interested in obtaining the minimum error independently of the value of the temperature. If we use M bits to represent the temperature values, it is expected that the fixed-point data is correct up to $2^{\beta-M}$; however the obtained precision is lower due to the fact that the operation with data with errors introduces new errors, as been said previously.

The choice of the wordlength depends strongly on the applications, in our case the criteria employed to fix the bitlength are:

1. The error introduced operating with fixed-point arithmetic should not mask the thermal differences between soil and mine. These differences varies from several degrees down to half degree depending on the period of the day and the depth at which the mine is buried.

2. In order to reduce the computing time it is critical to optimize the bus access to the memory. It is also important to reduce the critical path to increase the overall frequency of the system.

3. The area occupied by the circuit is an important issue; although in our case this is not crucial as the overall system occupies a small portion of the FPGA, as will be seen later.

It is necessary to chose the appropriate wordlength to represent temperatures, to this aim a trade-off that takes into account the occupied area, the speed and the precision has been

introduced:

$$\Gamma = \frac{N_T}{f \cdot A \cdot AE} \tag{3.5}$$

where Γ is the parameter that represents the trade-off for our application, f is the frequency at which the system is capable to work, A is the occupied area, N_T is is the number of temperature values that can be stored on a memory address and AE is the absolute error. The remainder parameters have been coded with an excess of bits ensure that the computations are correctly performed. This study has been performed using the data from a heating/cooling experiment of a piece of soil with mine absence, whose setup is described in Chapter 4. The results from various wordlengths are collected in Table 3.5. In Fig. 3.12 we can see the variation of the trade-off with the number of bits employed for the temperature. It can be noted how 16 bits is the optimal choice, this is due to the fact that it offers enough precision without spending too much area. However, the most important issue, comparing to the representation with 17 or 18 bits, is that it allows to duplicate the number of temperature values that can be read on each cycle, taking into account that the memory banks of the RC2000 are 32 bits wide and that if the temperatures are represented with 17 bits or more only one value can be stored on each memory address. We could think of using 9 or 10 bits to represent the temperatures, thus with these wordlengths 3 temperature values could be stored on each memory address; however with these wordlengths the precision is dramatically degraded and it is not useful for our algorithm.

In Fig. 3.13 the temperature at a surface point in heating/cooling process is shown. It can be noted the effects of using different number of bits to represent the integer data and how the reduction in the number of bits degrades the results of the computations; meanwhile there is no a bigger improvement in the precision when the temperature bitwidth goes from 16 bits to 18 bits. In Chapter 4 we will introduce the experimental setups of the experiments considered here. In Fig. 3.14 we can see the evolution of the temperature for a surface point in a different heating/cooling process using various bitlengths; in this figure we can see the temperature evolution for a point situated upon a shallowly buried mine and another one with no mine underneath. As can be noted the differences using 14 and 12 bits make it impossible to distinguish between mine and soil due to the differences that produces the limited range of values. Using 16 bits the difference between double precision computation and integer computation is lower than 0.3 ℃; however using 14 bits the maximum difference grows up until 2 ℃; finally, using 12 bits the maximum difference is in the order of 6 ℃, unacceptable for the application. Thus, 16 bits are employed to represent temperatures, that constitute the largest amount of data. Using this wordlength

Table 3.5: Parameters that affect the trade-off of the system.

Wordlength (bits)	Max Absolute Error (℃)	Occupied area (Slices)	T_{min} (ns)
18	0.08	734	29
17	0.09	665	24
16	0.17	600	20
15	0.9	541	16
14	2.1	484	9
13	3.2	431	3
12	4.7	382	7
11	6.4	116	1

to represent temperature nodes two temperatures can be stored on each memory address (32 bits wide) which allows to download from each memory bank up two temperature values per clock cycle. Once we have adjusted temperature's wordlength the remainder parameters are adjusted in a similar way. The wordlength and the factor of conversion employed to represent the data are shown in Table 3.6. As can be seen, the number of bits used to represent the temperature of the air are the same of the temperatures of the soil as the range of variation of both temperatures are similar. Concerning to the physical parameters of the soil (F_0) and those associated to the boundary conditions ($8F_0R, 2AF_0S, 2F_0H$) it has to be taken into account that we only have to distinguish between mine and soil. With these wordlengths the differences between mine and soil can be detected. Once the simulation has finished, the original format of the data is recovered by applying an inverse transformation. In this way data represented in different ranges can be easily aligned for computation by shift operations.

3.4.2 Hardware architecture

As was previously pointed, in the design of the system it has to be taken into account that the required programmability in the number of nodes and their high number (hundreds of thousands), makes impossible to perform a complete mapping node-processing element. Considering this limitation and the data that can be simultaneously read from the memory banks, an architecture with 8 processing elements (PE), labeled as $PE0$-$PE7$ in Fig. 3.10,

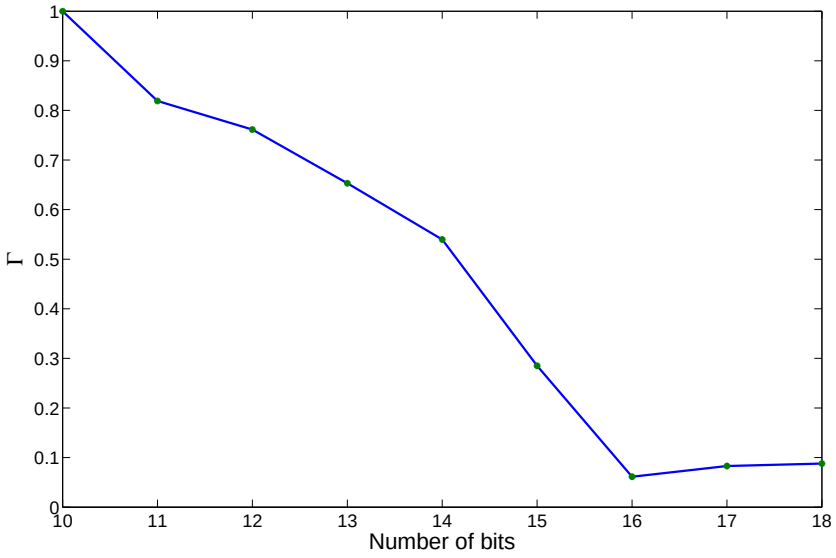
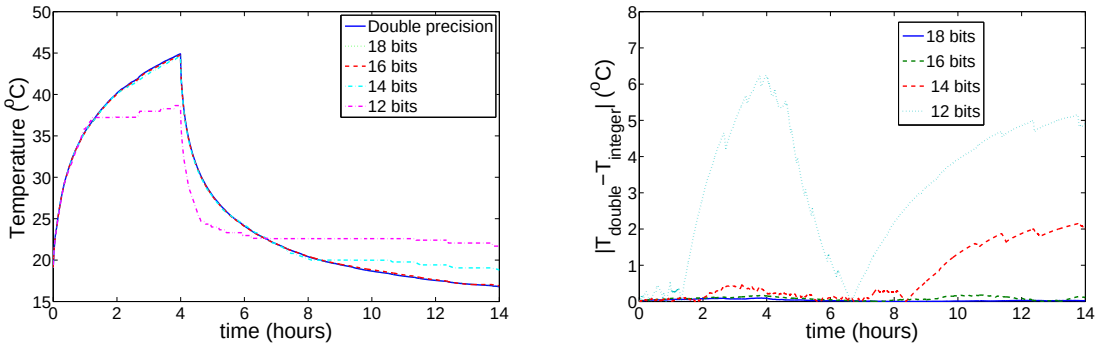


Figure 3.12: Variation of the trade-off with the number of bits.

has been proposed. Despite that the resources of the FPGA are not completely used, only 8 PEs have been implemented because the access to the memory banks, the bottleneck of the system, restricts the number of temperature values that can be read in parallel and therefore the number of PEs that can be continuously receiving data to be processed. If more memory banks were introduced in the system, more PEs could be added as the FPGA utilization is not a critical issue in this system, as will be seen later. Thus, the system is made up of 8 PEs, that compute the nodes' temperature updating, and a control unit (CU) composed of a control signal generator, an address generator that gives the memory addresses to read the temperates of the nodes being updated and their neighbors and the addresses where the updated values must be stored, and a control communications block that manages the ownership of the memory banks with the HOST, see Fig. 3.10.

The distribution of the data into the memory banks of the RC2000 card must be done carefully in order to obtain the highest possible performance. The eight memory banks are used to store node temperature values, all of them are used to read temperatures and to write the updated values, allowing in this way to read more temperature values in simultaneously. The data are distributed in z-layers, and consecutive layers are stored in consecutive memory banks; thus *MBank 0* stores layer 0 (z=0), layer 8 (z= 8) and so on, see Fig. 3.15. This allows reuse of data and an efficient way to read them from memory in the minimum possible number of clock cycles. All memory banks are divided in two



(a) Surface temperature (b)

Figure 3.13: Simulations using different bitlengths and differences between double and integer arithmetic data.

Table 3.6: Empirically obtained bitlengths and scaling factor.

Parameter	Wordlength	β
F_0	10	10
$8 \cdot F_0 \cdot R$	8	39
$2 \cdot A \cdot F_0 \cdot S$	10	18
$2 \cdot F_0 \cdot H$	8	14
$qsun$	17	7
$T_{i,j,k}, T_{air}$	16	9
T_{air}^3	16	-10

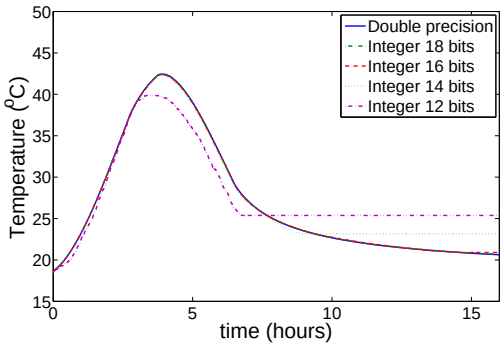
identical halves, one half is used to read temperatures and the other to write the updated values. Once an iteration has finished the two halves interchange their roles.

The way in which the data is read from memory banks is illustrated in Fig. 3.15. Each PE only receives temperature values corresponding to the nodes it updates and their neighbors. It can be noted that each PEk (k=0 ...7) updates the temperatures of the nodes stored in the corresponding memory bank $MBank\ k$ (k=0...7). As can be seen, only four clock cycles are necessary to read the data required to perform the updating due to the fact that the temperatures being updated and the previous neighbors have been loaded during the update of the previous nodes. On the other side, the parallelism of the system is also increased, allowing the parallel update of sixteen temperature values (two on each PE):

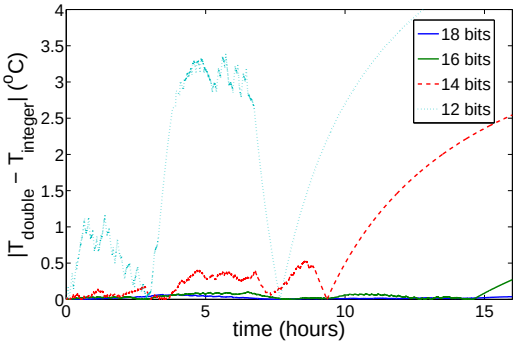
$T_{i,j,k}$, $T_{i+1,j,k}$; $T_{i,j,k+1}$, $T_{i+1,j,k+1}$; ... $T_{i,j,k+7}$, $T_{i+1,j,k+7}$. Given the capacity of the memory banks of the RC2000 card, simulations with up to two million nodes can be computed.

3.4.3 Processing element

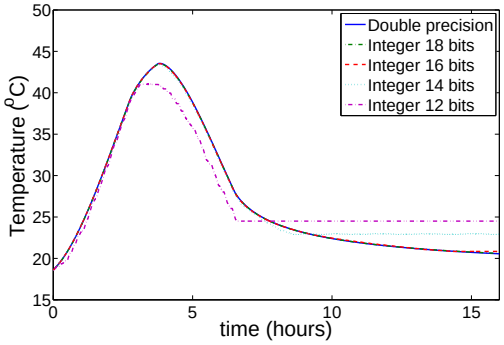
We will now make a detailed description of the PE that deals with the FD-TD set of equations. In the system there are 8 PE, that perform the nodes' temperature updating. Each PE follows the structure of Eqs. (2.36), (2.42), which are repeated here for the sake



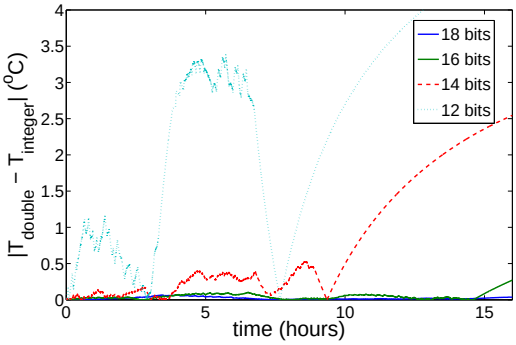
(a) Soil temperature without mine underneath



(b) Temperature differences (soil)



(c) Soil temperature with mine underneath at 3 mm



(d) Temperature differences (mine)

Figure 3.14: Simulations using different bitlengths and differences between double and integer arithmetic data for soil and mine.

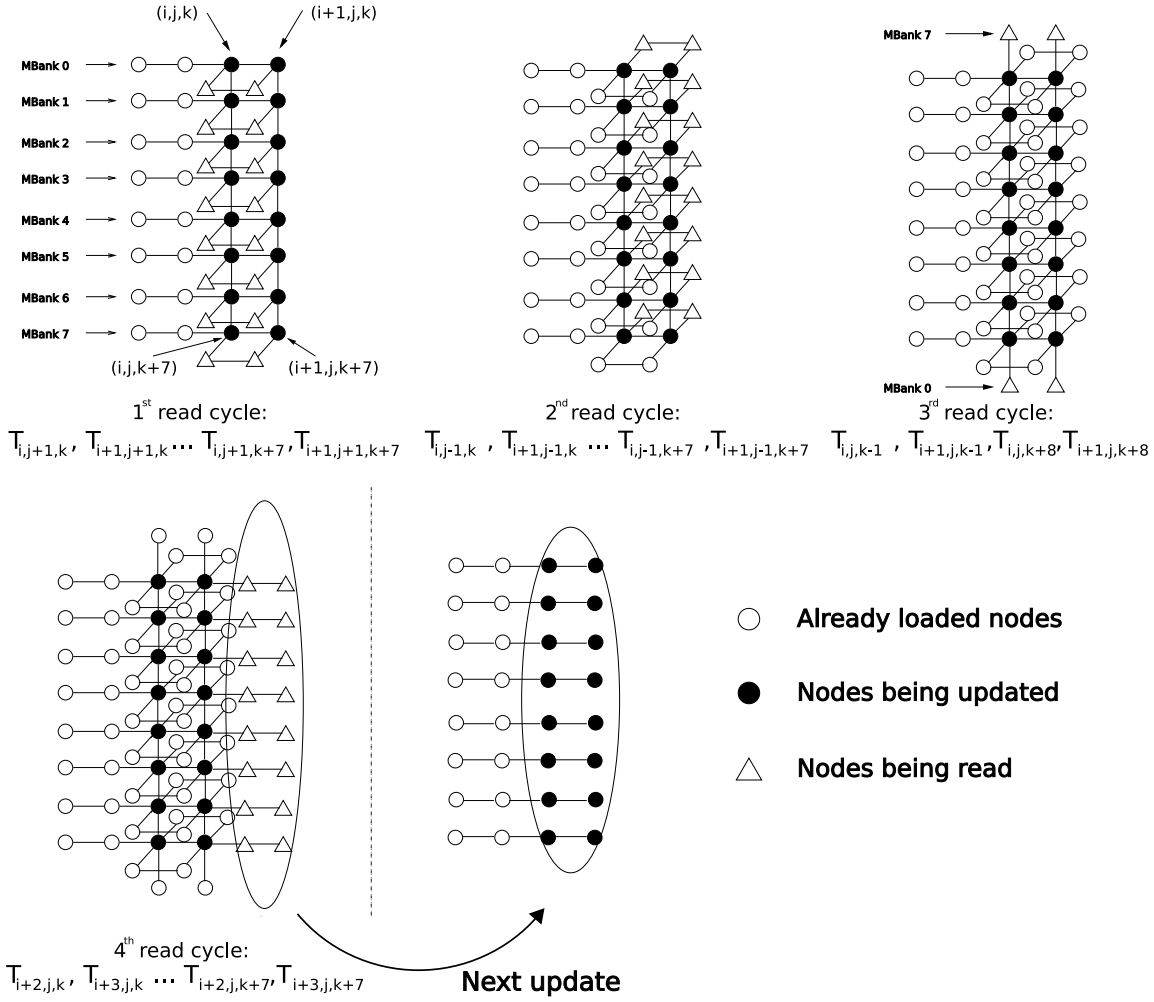


Figure 3.15: Scheme used to load data from memory. Clock cycles to read the necessary data to update the temperatures of nodes *(i,j,k)*, *(i+1,j,k)*... *(i,j,k+7)*, *(i+1,j,k+7)*.

of clarity:

$$\begin{aligned} T^{m+1}_{i,j,1} =& (1-6F_0)T^m_{i,j,1} + 2\epsilon_{sun}F_0 Sq^m_{sun} + 2F_0 H(T_{air} - T^m_{i,j,1}) + \\ & F_0(T^m_{i+1,j,1} + T^m_{i-1,j,1} + T^m_{i,j+1,1} + T^m_{i,j-1,1} + 2T^m_{i,j,2}) + \\ & 8F_0 R T^3_{air}(T_{air} - T^m_{i,j,1}) \end{aligned} \tag{3.6}$$

$$T^{m+1}_{i,j,k} = (1-6F_0)T^m_{i,j,k} + F_0(T^m_{i+1,j,k} + T^m_{i-1,j,k} + T^m_{i,j+1,k} + T^m_{i,j-1,k} + T^m_{i,j,k+1} + T^m_{i,j,k-1}) \tag{3.7}$$

These equations can be rewritten as:

$$\begin{aligned} T^{m+1}_{i,j,1} =& T^m_{i,j,1} + F_0\left(\sum_{neighbors1}(T^m_{neighbors1} - T^m_{i,j,1}) + 2(T^m_{i,j,2} - T^m_{i,j,1})\right) + \\ & 2\epsilon_{sun}F_0 Sq^m_{sun} + (2F_0 H + 8F_0 R T^3_{air})(T_{air} - T^m_{i,j,1}) \end{aligned} \tag{3.8}$$

$$T_{i,j,k}^{m+1} = T_{i,j,k}^{m} + F_0 \sum_{neighbors2} (T_{neighbors2}^{m} - T_{i,j,l}^{m}) \tag{3.9}$$

where $neighbors1$ in Eq. (3.8) makes reference to the nodes located in the same z-layer of node *(*i,j,1) and $neighbors2$ in Eq. (3.9) refers to all neighborhood of node *(i,j,k)*. Therefore, a PE is made up of adders, subtractors, multipliers and memory elements. In Fig. 3.16 the schematic corresponding to $PE1$-$PE6$ is shown; the schematics corresponding to $PE0$ and $PE7$ can be seen, respectively, in Figs. 3.17-3.18. We can distinguish four functional blocks: an *Input* stage, two sets of adders/subtractors labeled as *SUB* and *ACC*, and an output stage labeled as *Mult-Output*. $PE0$ and $PE7$ have a slight difference with the rest of the PEs in the configuration of the registers in the *Input* stage, as they receive data in the third read cycle, see Fig. 3.15. $PE0$ receives the temperatures of the north neighbors, see Fig. 3.17, and $PE7$ receives the temperatures of the south neighbors, see Fig. 3.18. Meanwhile, $PE1$-$PE6$ receive the temperatures from the upper and lower neighbors in the first read cycle. Additionally, $PE0$ has an extra circuitry that deals with the boundary conditions of the surface nodes, see Fig. 3.19.

Using the external ZBT RAM memory banks to store the data introduces the bottleneck of accessing a maximum of eighth memory locations simultaneously. Therefore, our aim was to optimize the structure so as to minimize data transfers and keep the processing hardware constantly busy using the data at the maximum rate available. A computing architecture was designed which is capable of updating several nodes in parallel. The choice of which nodes should be updated in parallel is based on maximizing reuse of the data to be fetched every time that new nodes are taken into account. This is related with the way in which the data is stored in the ZBT RAMs, as was described previously.

The core of the PE is composed of the two sets of adders/subtractors, labeled as *SUB* and *ACC* in Figs. 3.16-3.18. The data read on each clock cycle (temperatures of the nodes to be updated and/or their neighbors) are stored in the memory elements of the input stage. Next, the set of subtractors *SUB* performs the subtractions of the temperatures being updated with their neighbors. The accumulation of these differences takes place in the set of adders/subtractors labeled as *ACC*. Finally, in the output stage the remainder operations to complete Eqs. (3.8)-(3.9) are performed, obtaining the updated value of the temperature. The circuit includes a set of multiplexors which control the data flow.

The elements used in the design of the PEs have been developed using as hardware description language VHDL, (VHDL, 1987). The adders/subtracters were implemented as carry ripple adders to save area resources on the FPGA although in the end this was not

a limitation. We have used the multipliers of the Virtex-II to improve the performance with respect to one handmade, (Gschwind and Salapura, 1995), and to save area resources (slices). The memory elements are implemented as D flip-flops, which have an $enable - write$ signal, allowing us to control when the data can or can not be written on it.

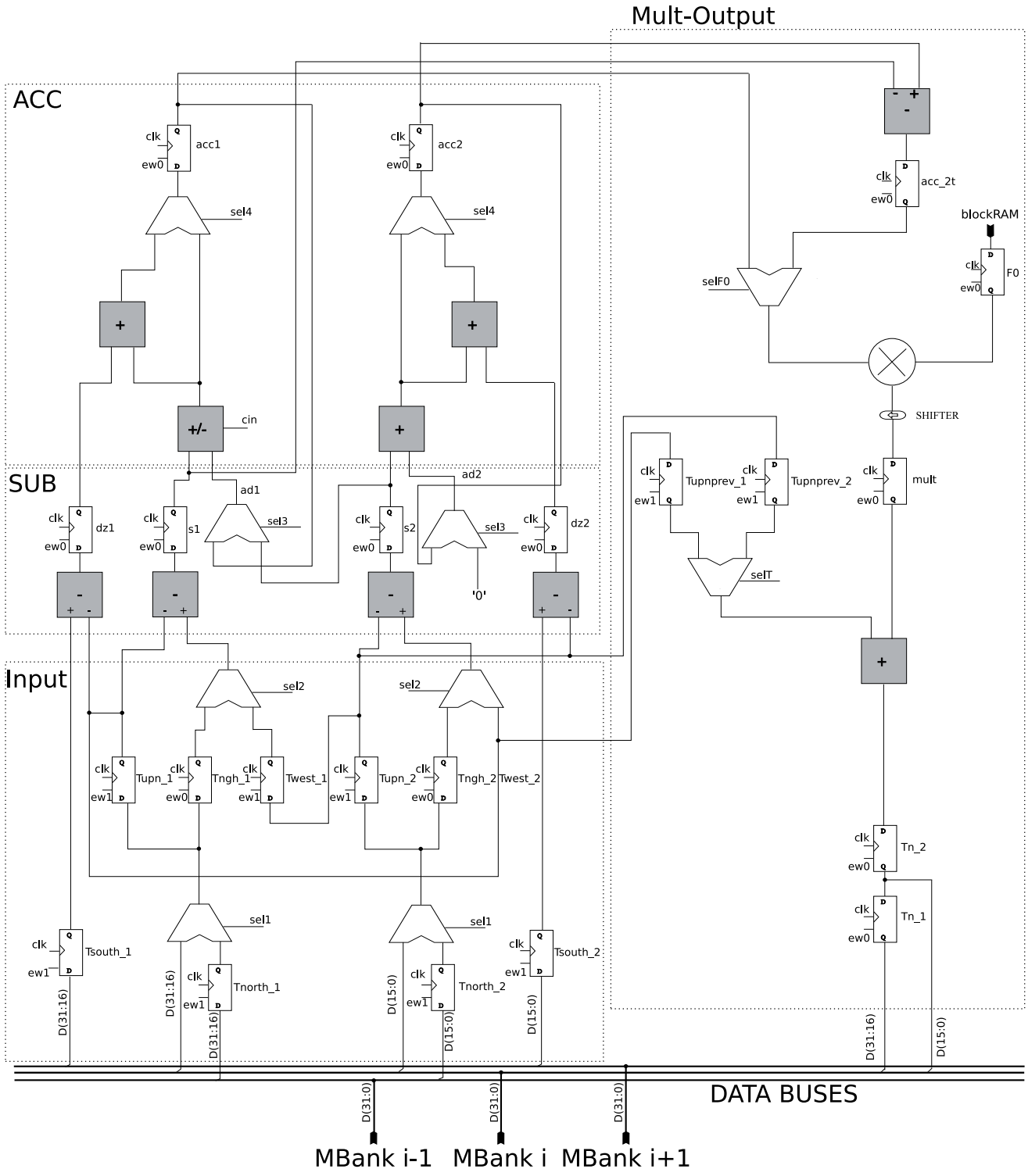


Figure 3.16: Processing element schematic: $PE1$... $PE6$.

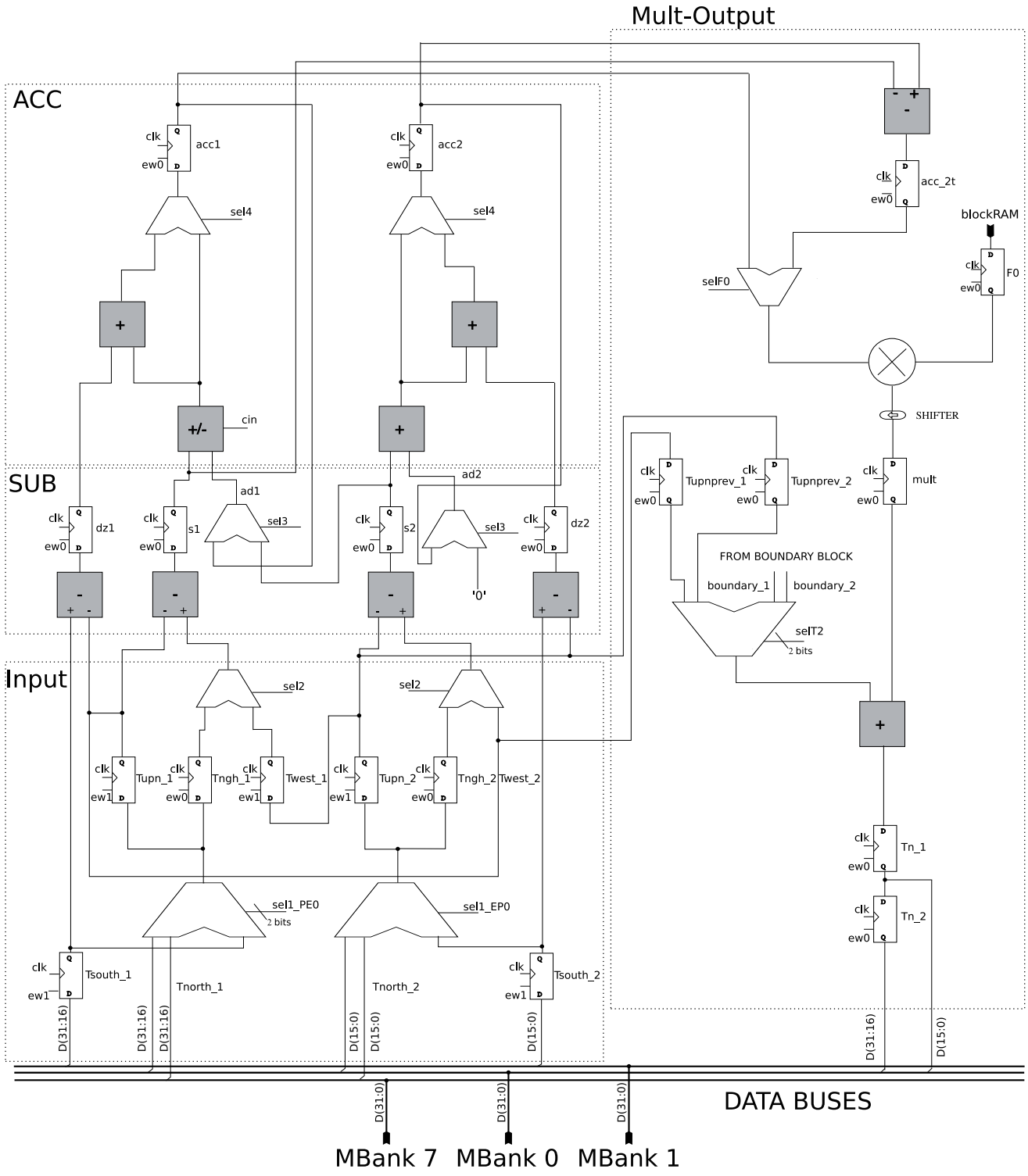


Figure 3.17: Processing element schematic $PE0$.

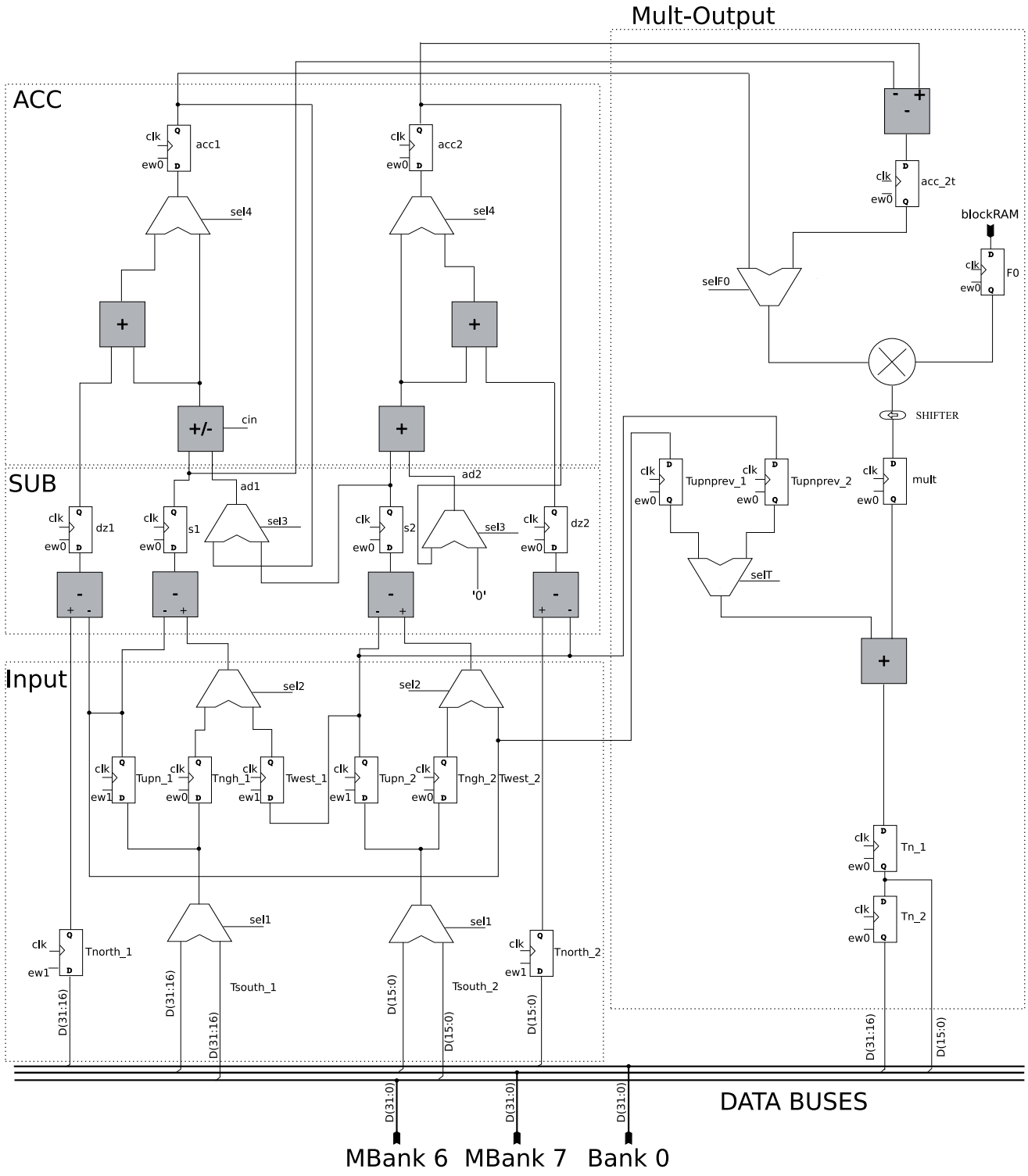


Figure 3.18: Processing element schematic $PE7$.

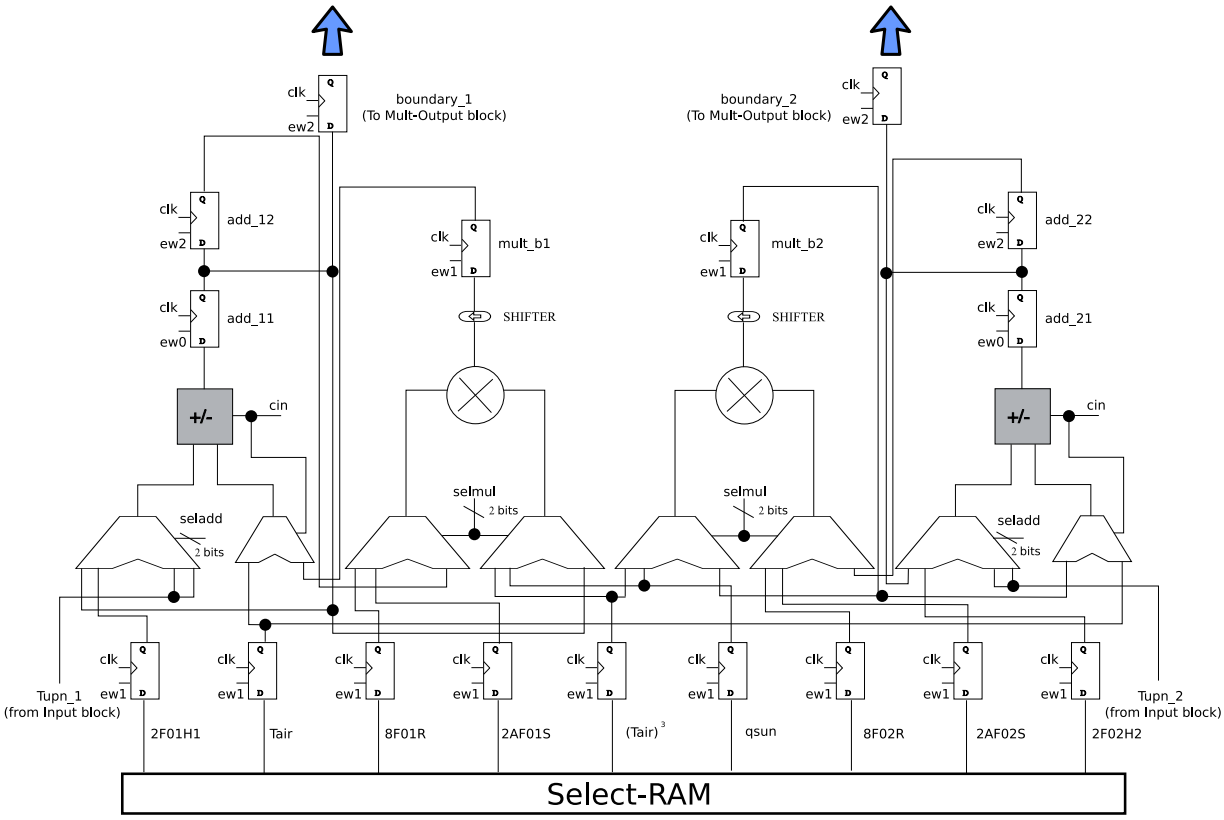


Figure 3.19: Boundary block: circuit schematic that deals with the boundary conditions.

3.4.4 Control Unit

As can be seen in Fig. 3.10, the control unit (CU) of the PE is made up of an *Index Generator* (IG), an *Address Decoder* (AD), a *Control Signal Generator* (CSG) and a *Control Communications* (CC) block. The core of the CU is a counter module $n_x \cdot n_y \cdot (n_z - 1)$ which provides the indexes (i,j,k) of the nodes that must be read/write from/to the memory banks in each clock cycle, in the IG block. From these indexes the AD generates the memory addresses of the corresponding points of the volume in the memory banks (the point under study and its neighbors). This decoding step is performed mapping the 3-D temperature matrix into a 1-D array using,

$$Address = i + jn_x + kn_xn_y \tag{3.10}$$

The CU also generates the control signals, in the CSG block, required by the PE, i.e.: *ew0*, *ew1*, *sel1*, *sel1_PE0*, *sel2*, *sel3*, *sel4*, *selF0*, *selT*, *selT2*, *cin*. The control signals *ew0* and *ew1* act as enable signals of the registers in the PE; whereas *sel1*, *sel1_PE0*, *sel2*,

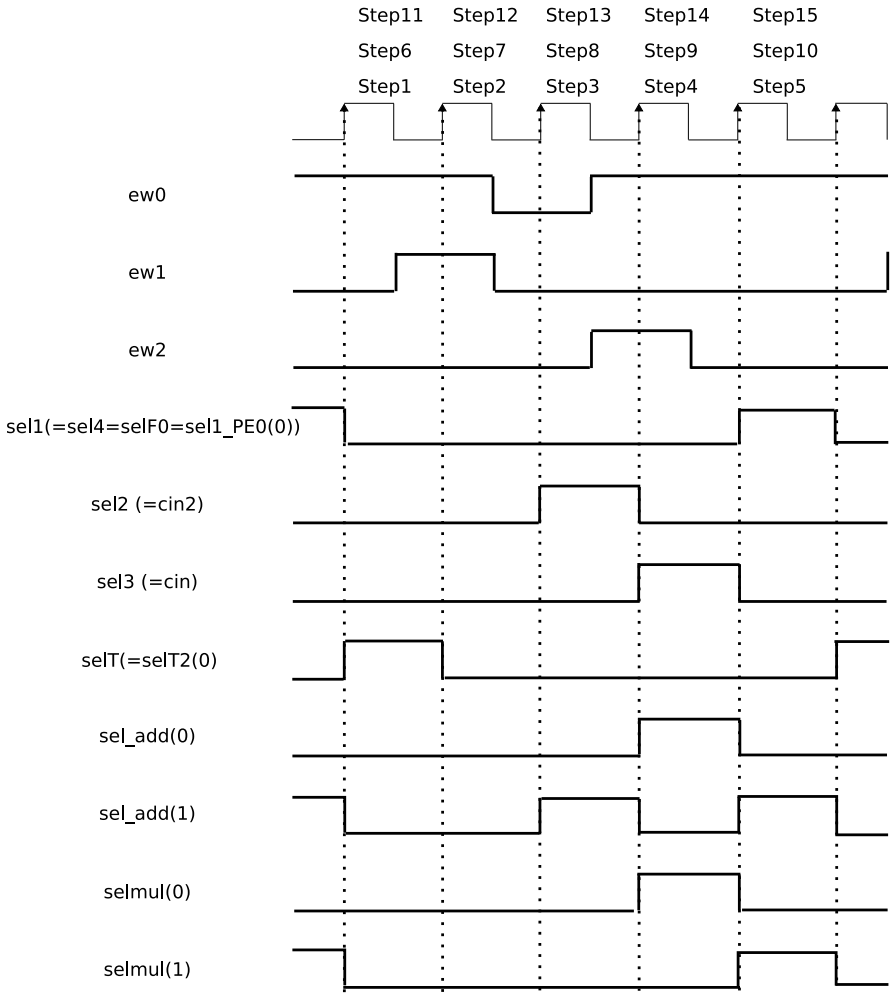


Figure 3.20: Chronogram of the control signals.

sel3, *sel4*, *selF0*, *selT* and *selT2* select one of the channels PE' multiplexors. Finally *cin* selects whether in the adder/subtractor of the ACC block is performed an addition or a subtraction. The chronogram of the control signals is shown in Fig. 3.20. The signals *sel1_PE0* and *selT2* control whether $PE0$ is processing an internal node or a surface node. In Fig. 3.20 we can see the least significant bit (LSB) of these two control signals, the most significant bit (MSB) is set to 0 when an internal node is being processed and it is set to 1 when a surface node is being processed.

The CC block manages the communications with the memory banks and with the HOST, avoiding bank utilization conflicts. It indicates to the HOST when the processing has finished or whether the HOST can get intermediate results from the memory banks.

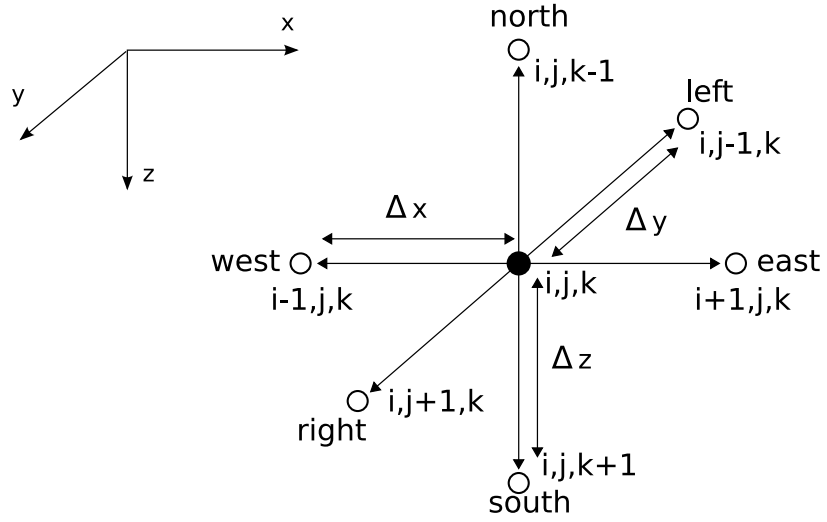


Figure 3.21: Considered neighbors to perform the updating process.

3.4.5 Data flow

This section describes the algorithm data flow with the help of Figs. 3.16-3.26 and Tables 3.7-3.18. Figs. 3.16, 3.17, 3.18 and 3.19 show the schematic of $PE1 - PE6$, $PE0$, $PE7$ and *Boundary block*, respectively. The chronogram of the control signals can be seen in Fig. 3.20. In Fig. 3.21 the considered neighbors of a node involved in the updating process are indicated. Fig. 3.22 shows the clock cycles necessary to update some pairs of nodes' temperature and the parallel processing that is being performed; access operations to the memory banks are also indicated. Figs. 3.23, 3.24, 3.25 show the operations performed with the memory banks, i.e., the read/write operations in *PEk* (k=1 ... 6), *PE0* and *PE7*, respectively; Fig. 3.26 shows the read/write operations to the memory banks when *PE0* is updating a surface node.

We will describe the data flow in the processing element PEk (k=0,...7) and we will start describing the steps needed to update the temperature of nodes (i,j,k) and (i+1,j,k)[1]. Then we will introduce the overlapped operations that are performed in parallel to update different nodes. We will consider as time origin the clock cycle in which the temperatures of the nodes *(i,j,k)* and *(i+1,j,k)* are read[2], i.e., *Step1* in Figs. 3.22-3.26.

[1]Even though the third index used to numerate the nodes, k, takes values from 1 to n_z, here it is redefined following the expression (k-1) = k *mod* 8, where k is the index that refers to the PEk. Thus, the index k of the nodes corresponds to the index k of the PE.

[2]The temperature reading of nodes *(i,j,k)* and *(i+1,j,k)* corresponds with the third reading cycle corresponding to the temperature update of the nodes *(i-2,j,k)* and *(i-1,j,k)*

As was previously said, the updating of nodes *(i,j,k)* and *(i+1,j,k)* begins in *Step1* and, in this clock cycle, their associated temperature values, $T^m_{i,j,k}$ and $T^m_{i+1,j,k}$, are stored in the registers *Tupn_1* and *Tupn_2* (see Figs. 3.16-3.18). At the same time, each processing element PEk (k=1,...6), in addition to the temperatures $T^m_{i,j,k}$ and $T^m_{i+1,j,k}$, receives from the corresponding memory banks the temperatures of the north and south neighbors ($T^m_{i,j,k-1}$, $T^m_{i+1,j,k-1}$ and $T^m_{i,j,k+1}$, $T^m_{i+1,j,k+1}$), which are stored in their corresponding registers (*Tnorth_1*, *Tnorth_2* and *Tsouth_1*, *Tsouth_2*) for their use in subsequent clock cycles, see Figs. 3.16,3.23. In addition to the temperatures whose updating begins, $PE0$ receives the data only from the south neighbor, see Fig. 3.25 and, similarly, $PE7$ only from the north one, see Fig. 3.24. Therefore, the structure of the input registers of these two processing elements differs from that of the rest, see Figs. 3.17, 3.18.

During *Step2* the system writes two temperatures that have been already updated ($T^m_{i-3,j,k}$, $T^m_{i-4,j,k}$) and the temperature updating process of nodes *(i,j,k)* and *(i+1,j,k)* is temporarily stopped, as seen in Figs 3.23-3.26. In the following clock cycles (*Step3* and *Step4* in Figs. 3.22-3.26) the temperatures of their right ($T^m_{i,j+1,k}$, $T^m_{i+1,j+1,k}$) and left ($T^m_{i,j-1,k}$, $T^m_{i+1,j-1,k}$) neighbors are read from memory and stored in the registers *Tngh_1* and *Tngh_2*. In *Step5* only $PE0$ and $PE7$ receive data; $PE0$ receives the temperature of the north neighbor from *MBank* 7, see Fig. 3.25, when it process an internal node, whereas when a surface node is being processed it does not read anything. In this clock cycle $PE7$ reads the temperature of its south neighbor from *MBank* 0, see Fig. 3.24. These values are stored in their corresponding registers (*Tngh_1* and *Tngh_2*). In this same clock cycle PEk (k=1,...6) transfer the values of the registers *Tnorth_1* and *Tnorth_2* to the registers *Tngh_1* and *Tngh_2* to be used in the next clock cycle.

The data read in one clock cycle are already available to operate with them in the next clock cycle. Thus, in *Step3* the differences from the west (*s1* and *s2*) and south (*dz1* and *dz2*) neighbors of $T^m_{i,j,k}$, $T^m_{i+1,j,k}$ are computed, except for *PE7* (as it has not yet received the data corresponding to its south neighbor). The temperature of the west neighbor was read in previous clock cycles, while the rest of the data was read in *Step1*. In the *SUB* block, during clock cycles *Step4*-*Step6* the differences between the temperatures of the nodes being updated (*Tupn_1* and *Tupn_2*) and those of the neighbors read in the previous clock cycle (*Tngh_1* and *Tngh_2*) are calculated and stored in the registers *s1* and *s2*. These differences are available to the *ACC* block in the next clock cycle, where the contributions of the different neighbors are accumulated, for the sake of implementing Eqs. (3.8)-(3.9). Thus, the update of the temperatures of nodes *(i,j,k)* and *(i+1,j,k)* in the *ACC* block begins in *Step4*. The accumulators *acc1* and *acc2* are initialized with the

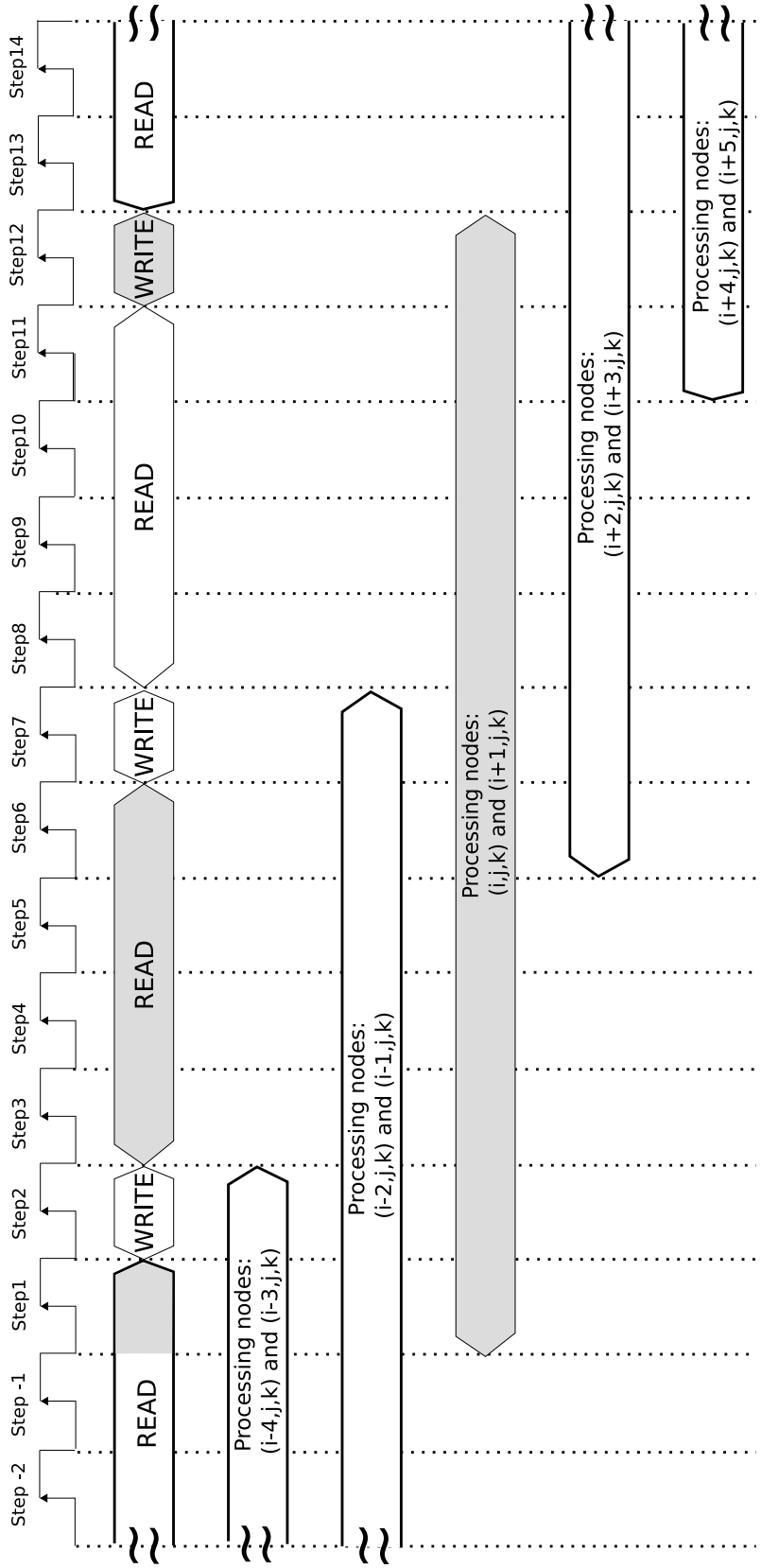


Figure 3.22: Temporal diagram of the read/write access to the memory and the clock cycles needed to update temperature nodes. In the clock cycles emphasized in grey operations to update the temperatures of nodes (i,j,k) and $(i+1,j,k)$ are performed.

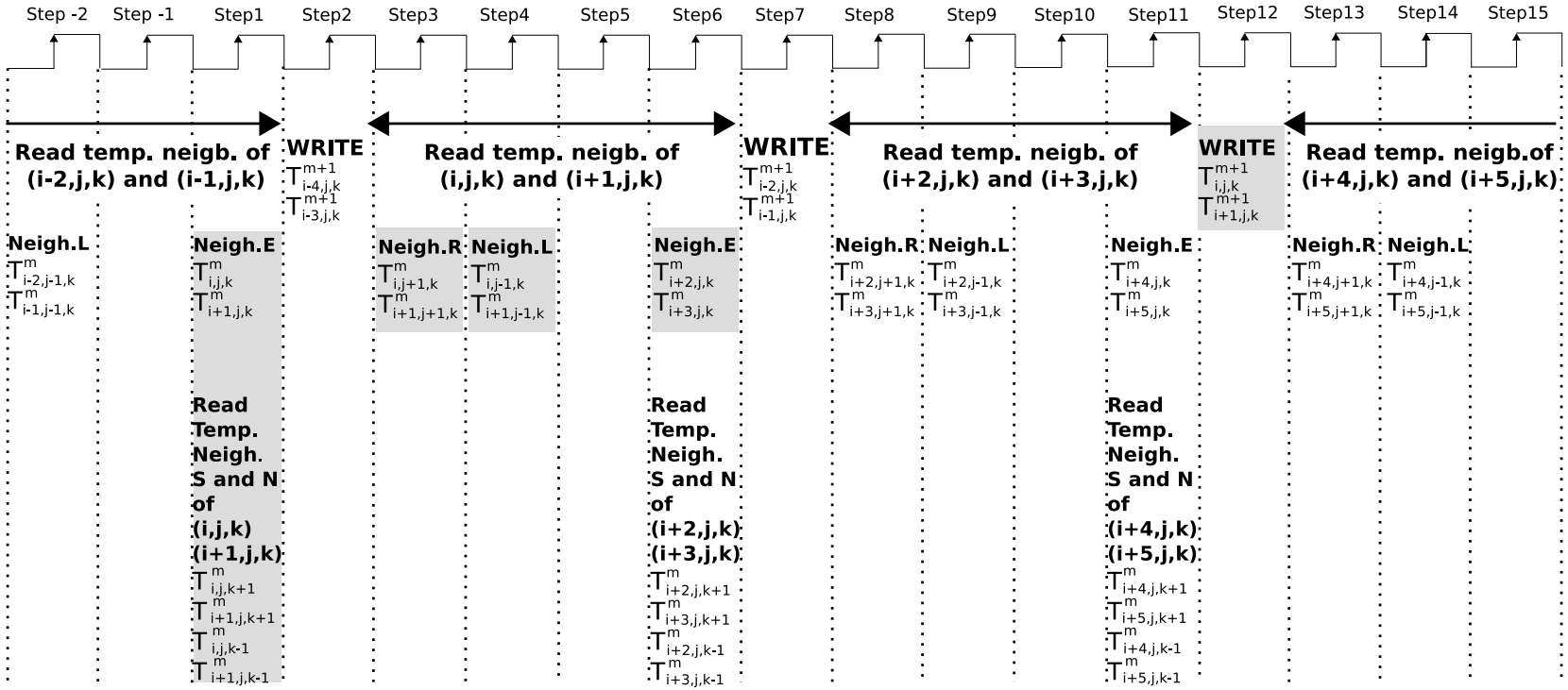


Figure 3.23: Scheme of the read/write operations from/to the memory banks by PEk (k=1...6). In the clock cycles emphasized in grey operations to update the temperatures of nodes *(i,j,k)* and *(i+1,j,k)* are performed.

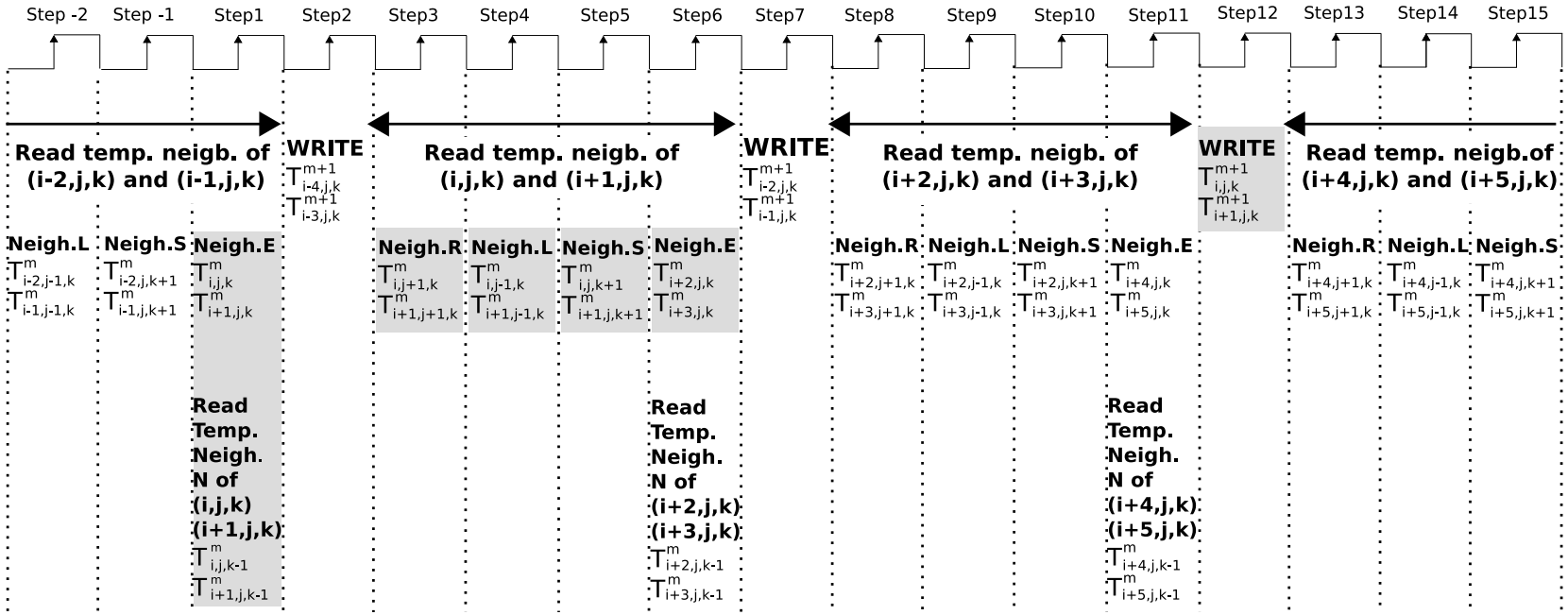


Figure 3.24: Scheme of the read/write operations from/to the memory banks by $PE7$. In the clock cycles emphasized in grey operations to update the temperatures of nodes *(i,j,k)* and *(i+1,j,k)* are performed.

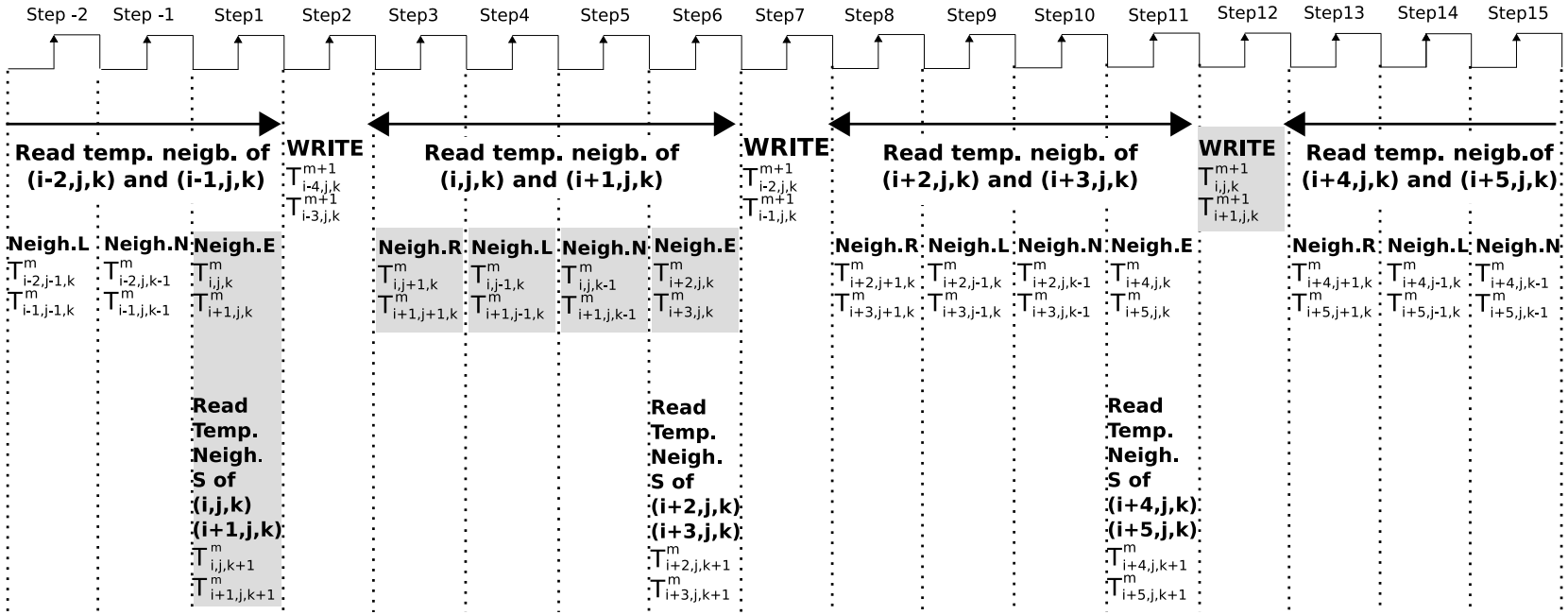


Figure 3.25: Scheme of the read/write operations from/to the memory banks by $PE0$ when it is processing internal nodes. In the clock cycles emphasized in grey operations to update the temperatures of nodes *(i,j,k)* and *(i+1,j,k)* are performed.

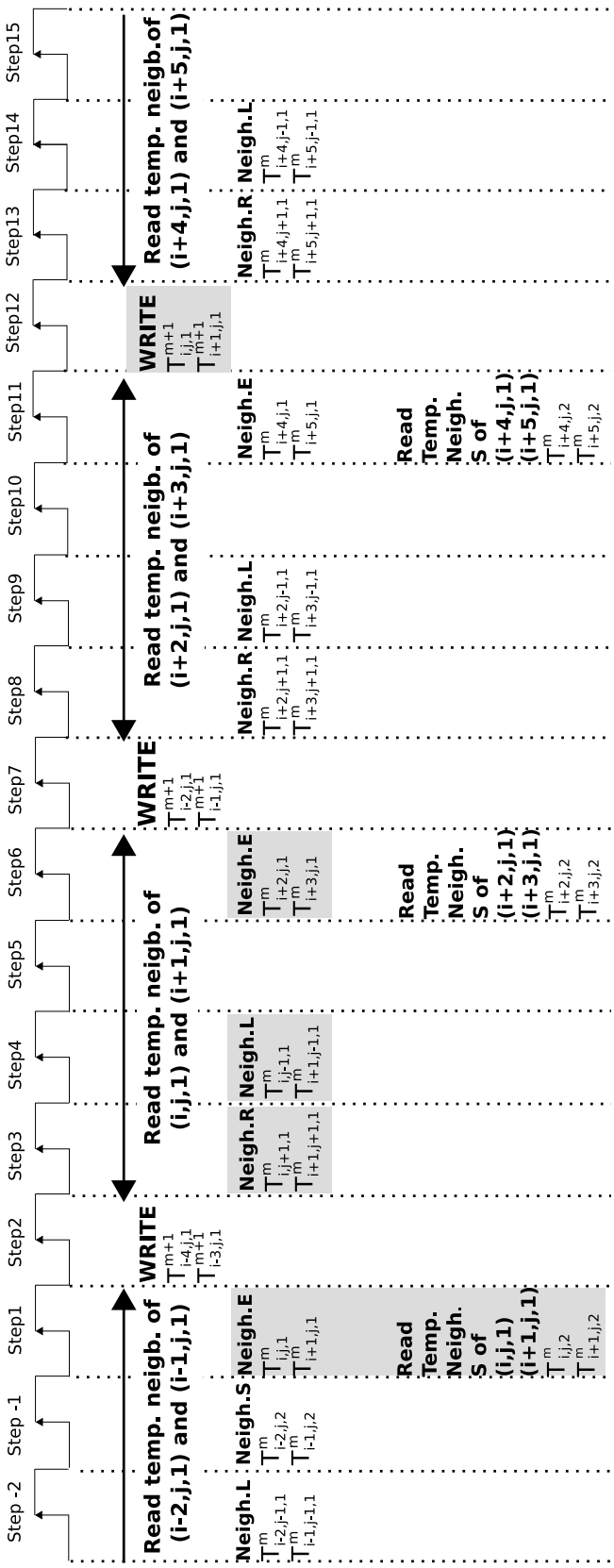


Figure 3.26: Scheme of the read/write operations from/to the memory banks by $PE0$ when it is processing surface nodes. In the clock cycles emphasized in grey operations to update the temperatures of nodes *(i,j,1)* and *(i+1,j,1)* are performed.

temperature differences with the east and west neighbors in the case of the *(i,j,k)* node and only with that of the west neighbor for the *(i+1,j,k)* node, obtained in *Step3*. In the following cycles, from *Step5* to *Step8*, these accumulators collect the contributions from the rest of the neighbors, see Figs 3.22-3.26.

In *Step5* the temperatures of all neighbors of node *(i,j,k)* have been already read, while for node *(i+1,j,k)* the temperature of its east neighbor, *(i+2,j,k)*, is not yet available and it will be read in *Step6*, which represents the 4th read cycle associated to the updating of the pair *(i,j,k)*, *(i+1,j,k)*, as seen in Fig. 3.15. In *Step6* a new pair of temperatures are loaded from memory in order to be updated and they are stored in registers *Tupn_1* and *Tupn_2*. As these registers contain the temperatures of nodes *(i,j,k)* and *(i+1,j,k)*, which are still necessary, their content is transferred to the registers *Tupnprev_1* and *Tupnprev_2* of the *Mult-Output* block. In *Step7* the updating process of two other temperature values is finished and their values are written to the memory banks and, therefore, the updating of nodes *(i,j,k)* and *(i+1,j,k)* is stopped again during this clock cycle, see Figs. 3.23-3.26.

In *Step8* the accumulator *acc1* has collected all neighbors' contributions of the node *(i,j,k)*; while all contributions for node *(i+1,j,k)* will be ready in *Step9* in the *acc_2t* accumulator. It is in the *Mult-Output* block where, during cycles *Step8* to *Step11*, the updated values $T_{i,j,k}^{m+1}$ and $T_{i+1,j,k}^{m+1}$ are obtained. To this aim, the values $F_0(i,j,k)$ and $F_0(i+1,j,k)$ are read from the block RAM of the FPGA and the operations required to complete Eq. (3.7) are performed: the multiplication of the corresponding value of F_0 by the accumulation of neighbors' contributions and the final addition of this multiplication with the initial value of the temperature (now in the registers *Tupnprev_1* and *Tupnprev_2*). In *Step11* the updated temperatures,$T_{i,j,k}^{m+1}$ and $T_{i+1,j,k}^{m+1}$, are already available and they will be loaded to the corresponding memory bank in the next clock cycle, *Step12*, see Figs. 3.22-3.25.

The previous description has been focused on the temperature updating of nodes *(i,j,k)* and *(i+1,j,k)*. However, as was said, several temperatures are being updated in parallel on each PE. For example, in *Step1*, when the temperatures of nodes *(i,j,k)* and *(i+1,j,k)* are read, the temperature updating process of nodes *(i-4,j,k)* and *(i-3,j,k)* has just finished and the updated values will be written to memory in *Step2*. Also, in *Step1* the temperatures of all the neighboring nodes of *(i-2,j,k)* and *(i-1,j,k)* have been collected, and their situation is identical to that of nodes *(i,j,k)* and *(i+1,j,k)* in *Step6*. From *Step3* to *Step6* the temperature updating process of nodes *(i-2,j,k)* and *(i-1,j,k)* continues in the *Mult-Output* block, in parallel with the temperature updating of nodes *(i,j,k)* and *(i+1,j,k)* in the ACC

and *SUB* blocks. Thus, in *Step6* the temperatures of nodes *(i-2,j,k)* and *(i-1,j,k)* have been already updated and in *Step7* are written to the corresponding memory bank, see Figs. 3.22-3.26. In *Step6* two new temperature values to be updated are read, those of nodes *(i+2,j,k)* and *(i+3,j,k)* and from *Step8* to *Step11* the updating of their temperatures is being performed in the *ACC* and *SUB* blocks, in parallel with the temperature updating of nodes *(i,j,k)* and *(i+1,j,k)* in the *Mult-Output* block.

As can be noted, the temperature updating process of a pair of nodes requires a total of twelve clock cycles; this is the number of clock cycles from the reading of the temperature values until the new temperatures are stored in the memory banks, see Fig. 3.22. It also can be seen that four temperatures are being updated in parallel on each clock cycle on each PE and two updated temperatures are written every five clock cycles. This scheme can also be seen in Fig. 3.20, where it can be noted that the control signals have a period of four clock cycles, which corresponds with the time interval between the reading of two new temperature values to be updated on each PE. Tables 3.7-3.9, Tables 3.10-3.12, Tables 3.13-3.15 summarize the complete data flow on *PE1... PE6*, $PE7$ and $PE0$, when it process internal nodes, respectively.

When the temperature updating of a surface node is carried out, the process in essentially identical but additional circuitry is necessary to take into account the contributions of the boundary conditions. A surface node has not north neighbor, thus, in the third read cycle in Fig. 3.15, $PE0$ does not receive anything. Thus, the *Mult-Output* multiplexor of $PE0$ has 4 inputs, to take into account the data that comes from the *Boundary* block. Now we will show the operations performed in the *Boundary* block, see Fig. 3.19. Thus, during *Step*1 the $PE0$ loads from the select-RAM the values[3] of T_{air}, T_{air}^3, $8F_{01}R_1$ and $8F_{02}R_2$. During *Step*2 the *Boundary* block stops the processing because a pair of temperatures are written to the memory banks, see Fig. 3.26. During *Step*3 $PE0$ uploads from select-RAM the values of $2F_{01}H_1$, $2F_{02}H_2$, $2F_{01}\alpha_{sun1}S_1$, $2F_{02}\alpha_{sun2}S_2$ and q_{sun}. Additionally, during this clock cycle, the system performs the following operations in the adders/subtractors: $T_{air} - T_{i,j,1}^m$ and $T_{air} - T_{i+1,j,1}^m$, while in the multipliers the following operations are performed: $T_{air}^3 \cdot 8F_{01}R_1$ and $T_{air}^3 \cdot 8F_{02}R_2$. All the data needed to process the boundary conditions have been already loaded from select-RAM. During *Step*4 the following operations are performed in the adders/subtractors: $2F_{01}H_1 + 8T_{air}^3F_{01}R_1$ and $2F_{01}H_1 + 8T_{air}^3F_{02}R_2$; while the multipliers perform these operations: $2F_{01}\epsilon sun1S_1 \cdot q_{sun}$

[3]To simplify the equations we have used the following notation $F_{01} = F_0(i,j,1)$, $F_{02} = F_0(i+1,j,1)$, $8F_0(i,j,1)R(i,j,1) = 8F_{01}R_1$ and so on.

and $2F_{02}\epsilon sun2S_2 \cdot q_{sun}$. Finally, during $Step5$ the operations related with the boundary conditions are finished with the addition of the values coming from the adders/subtractors and the multipliers obtained in $Step4$. These value is stored until it is required in the *Mult-Output* block to finish the updating of the $(i, j, 1)$ and $(i + 1, j, 1)$ nodes' temperature. All these operations are summarized in Tables 3.19-3.20. Tables 3.16-3.18 show the complete data flow in $PE0$ when it process surface nodes.

Clk	Step 1	Step 2	Step 3	Step 4	Step 5
Action on the memory banks	*Read* $T^m_{i,j,k}$, $T^m_{i+1,j,k}$ $T^m_{i,j,k-1}$, $T^m_{i+1,j,k-1}$ $T^m_{i,j,k+1}$, $T^m_{i+1,j,k+1}$	*Write* $T^{m+1}_{i-4,j,k}$ $T^{m+1}_{i-3,j,k}$	*Read* $T^m_{i,j+1,k}$, $T^m_{i+1,j+1,k}$	*Read* $T^m_{i,j-1,k}$, $T^m_{i+1,j-1,k}$	
Load of Data in input registers	*New nodes to be updated: (i,j,k) and (i+1,j,k)* $Twest\_1 = Tupn\_2\ (T^m_{i-1,j,k})$ $Tupn\_1 = T^m_{i,j,k}$ $Tupn\_2 = T^m_{i+1,j,k}$ $Tngh\_1 = T^m_{i,j,k}$ $Tngh\_2 = T^m_{i+1,j,k}$ $Tnorth\_1 = T^m_{i,j,k-1}$ $Tnorth\_2 = T^m_{i+1,j,k-1}$ $Tsouth\_1 = T^m_{i,j,k+1}$ $Tsouth\_2 = T^m_{i+1,j,k+1}$		$Tngh\_1 = T^m_{i,j+1,k}$ $Tngh\_2 = T^m_{i+1,j+1,k}$	$Tngh\_1 = T^m_{i,j-1,k}$ $Tngh\_2 = T^m_{i+1,j-1,k}$	$Tngh\_1 = Tnorth\_1$ $Tngh\_2 = Tnorth\_2$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tsouth\_1 - Tupn\_1$ $dz2 = Tsouth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i-2,j,k):* *add contrib. L* $acc1 = acc1 + s1$ *Update node (i-1,j,k):* *add contrib. L* $acc2 = acc2 + s2$		*Update node (i-2,j,k):* *add contrib. N and S* $acc1 = acc1 + s1 + dz1$ *Update node (i-1,j,k):* *add contrib. N and S* $acc2 = acc2 + s2 + dz2$	*Update node (i-1,j,k):* *add contrib. E* $acc2t = acc2 - s1$ *Update node (i,j,k):* *contrib. E and W* $acc1 = s1 - s2$ *Update node (i+1,j,k):* *contrib. W* $acc2 = s2$	*Update node (i,j,k):* *add contrib. R* $acc1 = acc1 + s1$ *Update node (i+1,j,k):* *add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i-4,j,k):* *Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^{m+1}_{i-2,j,k})$ *Update node (i-3,j,k):* *add* $T^m_{i-3,j,k}$ $Tn\_2 = mult + Tunpprev\_2$ $(Tn\_2 = T^{m+1}_{i-3,j,k})$ *Update nodes (i-2,j,k):* *and (i-1,j,k):* $Tupnprev\_1 = Tupn\_1(T^m_{i-2,j,k})$ $Tupnprev\_2 = Tupn\_2(T^m_{i-1,j,k})$		*Update node (i-2,j,k):* *Load F0(i-2,j,k)* $F0 = F0(i-2, j, k)$	*Update node (i-2,j,k):* *mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i-1,j,k):* *load F0(i-1,j,k)* $F0 = F0(i-1, j, k)$	*Update node (i-2,j,k):* *add* $T^m_{i-2,j,k}$ $Tn\_2 = mult + Tupnprev\_1$ $(Tn\_2 = T^{m+1}_{i-2,j,k})$ *Update node (i-1,j,k):* *mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.7: Data flow on $PE1$... $PE6$ (I).

Clk	Step 6	Step 7	Step 8	Step 9	Step 10
Action on the memory banks	*Read* $T^m_{i+2,j,k}$, $T^m_{i+3,j,k}$ $T^m_{i+2,j,k-1}$, $T^m_{i+3,j,k-1}$ $T^m_{i+2,j,k+1}$, $T^m_{i+3,j,k+1}$	*Write* $T^{m+1}_{i-2,j,k}$ $T^{m+1}_{i-1,j,k}$	*Read* $T^m_{i+2,j+1,k}$, $T^m_{i+3,j+1,k}$	*Read* $T^m_{i+2,j-1,k}$, $T^m_{i+3,j-1,k}$	
Load of Data in input registers	*New nodes to be updated: (i+2,j,k) and (i+3,j,k)* $Twest\_1 = Tupn\_2$ $(T^m_{i+1,j,k})$ $Tupn\_1 = T^m_{i+2,j,k}$ $Tupn\_2 = T^m_{i+3,j,k}$ $Tngh\_1 = T^m_{i+2,j,k}$ $Tngh\_2 = T^m_{i+3,j,k}$ $Tnorth\_1 = T^m_{i+2,j,k-1}$ $Tnorth\_2 = T^m_{i+3,j,k-1}$ $Tsouth\_1 = T^m_{i+2,j,k+1}$ $Tsouth\_2 = T^m_{i+3,j,k+1}$		$Tngh\_1 = T^m_{i+2,j+1,k}$ $Tngh\_2 = T^m_{i+3,j+1,k}$	$Tngh\_1 = T^m_{i+2,j-1,k}$ $Tngh\_2 = T^m_{i+3,j-1,k}$	$Tngh\_1 = Tnorth\_1$ $Tngh\_2 = Tnorth\_2$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tsouth\_1 - Tupn\_1$ $dz2 = Tsouth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i,j,k): add contrib. L* $acc1 = acc1 + s1$ *Update node (i+1,j,k): add contrib. L* $acc2 = acc2 + s2$		*Update node (i,j,k): add contrib. N and S* $acc1 = acc1 + s1 + dz1$ *Update node (i+1,j,k): add contrib. N and S* $acc2 = acc2 + s2 + dz2$	*Update node (i+1,j,k): add contrib. W* $acc2t = acc2 - s1$ *Update node (i+2,j,k): contrib. E and W* $acc1 = s1 - s2$ *Update node (i+3,j,k): contrib. E* $acc2 = s2$	*Update node (i+2,j,k): add contrib. R* $acc1 = acc1 + s1$ *Update node (i+3,j,k): add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i-2,j,k): Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^{m+1}_{i-2,j,k})$ *Update node (i-1,j,k): add* $T^m_{i-1,j,k}$ $Tn\_2 = mult + Tunpprev\_2$ $(Tn\_2 = T^{m+1}_{i-1,j,k})$ *Update nodes (i,j,k) and (i+1,j,k):* $Tupnprev\_1 = Tupn\_1(T^m_{i-2,j,k})$ $Tupnprev\_2 = Tupn\_2(T^m_{i-1,j,k})$		*Update node (i,j,k): Load F0(i,j,k)* $F0 = F0(i,j,k)$	*Update node (i,j,k): mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i+1,j,k): load F0(i+1,j,k)* $F0 = F0(i+1,j,k)$	*Update node (i,j,k): add* $T^m_{i,j,k}$ $Tn\_2 = mult + Tupnprev\_1$ $(Tn\_2 = T^{m+1}_{i,j,k})$ *Update node (i+1,j,k): mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.8: Data flow on $PE1$... $PE6$ (II).

Clk	Step 11	Step 12	Step 13	Step 14	Step 15
Action on the memory banks	*Read* $T^m_{i+4,j,k}$, $T^m_{i+5,j,k}$ $T^m_{i+4,j,k-1}$, $T^m_{i+5,j,k-1}$ $T^m_{i+4,j,k+1}$, $T^m_{i+4,j,k+1}$	*Write* $T^{m+1}_{i,j,k}$ $T^{m+1}_{i+1,j,k}$	*Read* $T^m_{i+4,j+1,k}$, $T^m_{i+5,j+1,k}$	*Read* $T^m_{i+4,j-1,k}$, $T^m_{i+5,j-1,k}$	
Load of Data in input registers	*New nodes to be updated: (i+4,j,k) and (i+5,j,k)* $Twest\_1 = Tupn\_2$ $(T^m_{i+3,j,k})$ $Tupn\_1 = T^m_{i+4,j,k}$ $Tupn\_2 = T^m_{i+5,j,k}$ $Tngh\_1 = T^m_{i+4,j,k}$ $Tngh\_2 = T^m_{i+5,j,k}$ $Tnorth\_1 = T^m_{i+4,j,k-1}$ $Tnorth\_2 = T^m_{i+5,j,k-1}$ $Tsouth\_1 = T^m_{i+4,j,k+1}$ $Tsouth\_2 = T^m_{i+5,j,k+1}$		$Tngh\_1 = T^m_{i+4,j+1,k}$ $Tngh\_2 = T^m_{i+5,j+1,k}$	$Tngh\_1 = T^m_{i+4,j-1,k}$ $Tngh\_2 = T^m_{i+5,j-1,k}$	$Tngh\_1 = Tnorth\_1$ $Tngh\_2 = Tnorth\_2$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tsouth\_1 - Tupn\_1$ $dz2 = Tsouth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i+2,j,k): add contrib. L* $acc1 = acc1 + s1$ *Update node (i+3,j,k): add contrib. L* $acc2 = acc2 + s2$		*Update node (i+2,j,k): add contrib. N and S* $acc1 = acc1 + s1 + dz1$ *Update node (i+3,j,k): add contrib. N and S* $acc2 = acc2 + s2 + dz2$	*Update node (i+3,j,k): add contrib. W* $acc2t = acc2 - s1$ *Update node (i+4,j,k): contrib. E and W* $acc1 = s1 - s2$ *Update node (i+5,j,k): contrib. W* $acc2 = s2$	*Update node (i+4,j,k): add contrib. R* $acc1 = acc1 + s1$ *Update node (i+5,j,k): add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i,j,k): Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^{m+1}_{i,j,k})$ *Update node (i+1,j,k): add* $T^m_{i+1,j,k}$ $Tn\_2 = mult + Tunpprev\_2$ $(Tn\_2 = T^{m+1}_{i+1,j,k})$ *Update nodes (i+2,j,k) and (i+3,j,k):* $Tupnprev\_1 = Tupn\_1(T^m_{i+2,j,k})$ $Tupnprev\_2 = Tupn\_2(T^m_{i+3,j,k})$		*Update node (i+2,j,k): Load F0(i+2,j,k)* $F0 = F0(i+2, j, k)$	*Update node (i+2,j,k): mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i+3,j,k): load F0(i+3,j,k)* $F0 = F0(i+3, j, k)$	*Update node (i+2,j,k): add* $T^m_{i+2,j,k}$ $Tn\_2 = mult + Tupnprev\_1$ $(Tn\_2 = T^{m+1}_{i+2,j,k})$ *Update node (i+3,j,k): mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.9: Data flow on $PE1$... $PE6$ (III).

Clk	Step 1	Step 2	Step 3	Step 4	Step 5
Action on the memory banks	*Read* $T^m_{i,j,k}$, $T^m_{i+1,j,k}$ $T^m_{i,j,k-1}$, $T^m_{i+1,j,k-1}$	*Write* $T^{m+1}_{i-4,j,k}$ $T^{m+1}_{i-3,j,k}$	*Read* $T^m_{i,j+1,k}$, $T^m_{i+1,j+1,k}$	*Read* $T^m_{i,j-1,k}$, $T^m_{i+1,j-1,k}$	*Read* $T^m_{i,j,k+1}$, $T^m_{i+1,j,k+1}$
Load of Data in input registers	*New nodes to be updated: (i,j,k) and (i+1,j,k)* $Twest\_1 = Tupn\_2\ (T^m_{i-1,j,k})$ $Tupn\_1 = T^m_{i,j,k}$ $Tupn\_2 = T^m_{i+1,j,k}$ $Tngh\_1 = T^m_{i,j,k}$ $Tngh\_2 = T^m_{i+1,j,k}$ $Tnorth\_1 = T^m_{i,j,k-1}$ $Tnorth\_2 = T^m_{i+1,j,k-1}$		$Tngh\_1 = T^m_{i,j+1,k}$ $Tngh\_2 = T^m_{i+1,j+1,k}$	$Tngh\_1 = T^m_{i,j-1,k}$ $Tngh\_2 = T^m_{i+1,j-1,k}$	$Tngh\_1 = T^m_{i,j,k+1}$ $Tngh\_2 = T^m_{i+1,j,k+1}$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tnorth\_1 - Tupn\_1$ $dz2 = Tnorth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i-2,j,k): add contrib. L* $acc1 = acc1 + s1$ *Update node (i-1,j,k): add contrib. L* $acc2 = acc2 + s2$		*Update node (i-2,j,k): add contrib. N and S* $acc1 = acc1 + s1 + dz1$ *Update node (i-1,j,k): add contrib. N and S* $acc2 = acc2 + s2 + dz2$	*Update node (i-1,j,k): add contrib. E* $acc2t = acc2 - s1$ *Update node (i,j,k): contrib. E and W* $acc1 = s1 - s2$ *Update node (i+1,j,k): contrib. W* $acc2 = s2$	*Update node (i,j,k): add contrib. R* $acc1 = acc1 + s1$ *Update node (i+1,j,k): add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i-4,j,k): Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^{m+1}_{i-4,j,k})$ *Update node (i-3,j,k): add* $T^m_{i-3,j,k}$ $Tn\_2 = mult + Tunpprev\_2$ $(Tn\_2 = T^{m+1}_{i-3,j,k})$ *Update nodes (i-2,j,k) and (i-1,j,k):* $Tupnprev\_1 = Tupn\_1(T^m_{i-2,j,k})$ $Tupnprev\_2 = Tupn\_2(T^m_{i-1,j,k})$		*Update node (i-2,j,k): Load F0(i-2,j,k)* $F0 = F0(i-2, j, k)$	*Update node (i-2,j,k): mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i-1,j,k): load F0(i-1,j,k)* $F0 = F0(i-1, j, k)$	*Update node (i-2,j,k): add* $T^m_{i-2,j,k}$ $Tn\_2 = mult + Tupnprev\_1$ $(Tn\_2 = T^{m+1}_{i-2,j,k})$ *Update node (i-1,j,k): mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.10: Data flow on $PE7$ (I).

Clk	Step 6	Step 7	Step 8	Step 9	Step 10
Action on the memory banks	*Read* $T^m_{i+2,j,k}$, $T^m_{i+3,j,k}$ $T^m_{i+2,j,k-1}$, $T^m_{i+3,j,k-1}$	*Write* $T^{m+1}_{i-2,j,k}$ $T^{m+1}_{i-1,j,k}$	*Read* $T^m_{i+2,j+1,k}$, $T^m_{i+3,j+1,k}$	*Read* $T^m_{i+2,j-1,k}$, $T^m_{i+3,j-1,k}$	*Read* $T^m_{i+2,j,k+1}$, $T^m_{i+3,j,k+1}$
Load of Data in input registers	*New nodes to be updated: (i+2,j,k) and (i+3,j,k)* $Twest\_1 = Tupn\_2$ $(T^m_{i+1,j,k})$ $Tupn\_1 = T^m_{i+2,j,k}$ $Tupn\_2 = T^m_{i+3,j,k}$ $Tngh\_1 = T^m_{i+2,j,k}$ $Tngh\_2 = T^m_{i+3,j,k}$ $Tnorth\_1 = T^m_{i+2,j,k-1}$ $Tnorth\_2 = T^m_{i+3,j,k-1}$		$Tngh\_1 = T^m_{i+2,j+1,k}$ $Tngh\_2 = T^m_{i+3,j+1,k}$	$Tngh\_1 = T^m_{i+2,j-1,k}$ $Tngh\_2 = T^m_{i+3,j-1,k}$	$Tngh\_1 = T^m_{i+2,j,k+1}$ $Tngh\_2 = T^m_{i+3,j,k+1}$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tnorth\_1 - Tupn\_1$ $dz2 = Tnorth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i,j,k):* *add contrib. L* $acc1 = acc1 + s1$ *Update node (i+1,j,k):* *add contrib. L* $acc2 = acc2 + s2$		*Update node (i,j,k):* *add contrib. N and S* $acc1 = acc1 + s1 + dz1$ *Update node (i+1,j,k):* *add contrib. N and S* $acc2 = acc2 + s2 + dz2$	*Update node (i+1,j,k):* *add contrib. W* $acc2t = acc2 - s1$ *Update node (i+2,j,k):* *contrib. E and W* $acc1 = s1 - s2$ *Update node (i+3,j,k):* *contrib. E* $acc2 = s2$	*Update node (i+2,j,k):* *add contrib. R* $acc1 = acc1 + s1$ *Update node (i+3,j,k):* *add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i-2,j,k):* *Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^{m+1}_{i-2,j,k})$ *Update node (i-1,j,k):* *add* $T^m_{i-1,j,k}$ $Tn\_2 = mult + Tunpprev\_2$ $(Tn\_2 = T^{m+1}_{i-1,j,k})$ *Update nodes (i,j,k) and (i+1,j,k):* $Tupnprev\_1 = Tupn\_1(T^m_{i-2,j,k})$ $Tupnprev\_2 = Tupn\_2(T^m_{i-1,j,k})$		*Update node (i,j,k):* *Load F0(i,j,k)* $F0 = F0(i,j,k)$	*Update node (i,j,k):* *mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i+1,j,k):* *load F0(i+1,j,k)* $F0 = F0(i+1,j,k)$	*Update node (i,j,k):* *add* $T^m_{i,j,k}$ $Tn\_2 = mult + Tupnprev\_1$ $(Tn\_2 = T^{m+1}_{i,j,k})$ *Update node (i+1,j,k):* *mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.11: Data flow on $PE7$ (II).

Clk	Step 11	Step 12	Step 13	Step 14	Step 15
Action on the memory banks	*Read* $T^m_{i+4,j,k}$, $T^m_{i+5,j,k}$ $T^m_{i+4,j,k-1}$, $T^m_{i+5,j,k-1}$	*Write* $T^{m+1}_{i,j,k}$ $T^{m+1}_{i+1,j,k}$	*Read* $T^m_{i+4,j+1,k}$, $T^m_{i+5,j+1,k}$	*Read* $T^m_{i+4,j-1,k}$, $T^m_{i+5,j-1,k}$	*Read* $T^m_{i+4,j,k+1}$, $T^m_{i+5,j,k+1}$
Load of Data in input registers	*New nodes to be updated: (i+4,j,k) and (i+5,j,k)* $Twest\_1 = Tupn\_2\ (T^m_{i+3,j,k})$ $Tupn\_1 = T^m_{i+4,j,k}$ $Tupn\_2 = T^m_{i+5,j,k}$ $Tngh\_1 = T^m_{i+4,j,k}$ $Tngh\_2 = T^m_{i+5,j,k}$ $Tnorth\_1 = T^m_{i+4,j,k-1}$ $Tnorth\_2 = T^m_{i+5,j,k-1}$		$Tngh\_1 = T^m_{i+4,j+1,k}$ $Tngh\_2 = T^m_{i+5,j+1,k}$	$Tngh\_1 = T^m_{i+4,j-1,k}$ $Tngh\_2 = T^m_{i+5,j-1,k}$	$Tngh\_1 = T^m_{i+4,j,k+1}$ $Tngh\_2 = T^m_{i+5,j,k+1}$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tnorth\_1 - Tupn\_1$ $dz2 = Tnorth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i+2,j,k):* *add contrib. L* $acc1 = acc1 + s1$ *Update node (i+3,j,k):* *add contrib. L* $acc2 = acc2 + s2$		*Update node (i+2,j,k):* *add contrib. N and S* $acc1 = acc1 + s1 + dz1$ *Update node (i+3,j,k):* *add contrib. N and S* $acc2 = acc2 + s2 + dz2$	*Update node (i+3,j,k):* *add contrib. W* $acc2t = acc2 - s1$ *Update node (i+4,j,k):* *contrib. E and W* $acc1 = s1 - s2$ *Update node (i+5,j,k):* *contrib. E* $acc2 = s2$	*Update node (i+4,j,k):* *add contrib. R* $acc1 = acc1 + s1$ *Update node (i+5,j,k):* *add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i,j,k):* *Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^m_{i,j,k})$ *Update node (i+1,j,k):* *add* $T^m_{i+1,j,k}$ $(Tn\_2 = mult + Tunpprev\_2)$ $(Tn\_2 = T^{m+1}_{i+1,j,k})$ *Update nodes (i+2,j,k): and (i+3,j,k):* $Tupnprev\_1 = Tupn\_1(T^m_{i+2,j,k})$ $Tupnprev\_2 = Tupn\_2(T^m_{i+3,j,k})$		*Update node (i+2,j,k):* *Load F0(i+2,j,k)* $F0 = F0(i+2, j, k)$	*Update node (i+2,j,k):* *mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i+3,j,k):* *load F0(i+3,j,k)* $F0 = F0(i+3, j, k)$	*Update node (i+2,j,k)* *add* $T^m_{i+2,j,k}$ $Tn\_2 = mult + Tupnprev\_1$ $(Tn\_2 = T^{m+1}_{i+2,j,k})$ *Update node (i+3,j,k):* *mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.12: Data flow on $PE7$ (III).

Clk	Step 1	Step 2	Step 3	Step 4	Step 5
Action on the memory banks	*Read* $T^m_{i,j,k}$, $T^m_{i+1,j,k}$ $T^m_{i,j,k+1}$, $T^m_{i+1,j,k+1}$	*Write* $T^{m+1}_{i-4,j,k}$ $T^{m+1}_{i-3,j,k}$	*Read* $T^m_{i,j+1,k}$, $T^m_{i+1,j+1,k}$	*Read* $T^m_{i,j-1,k}$, $T^m_{i+1,j-1,k}$	*Read* $T^m_{i,j,k-1}$, $T^m_{i+1,j,k-1}$
Load of Data in input registers	*New nodes to be updated: (i,j,k) and (i+1,j,k)* $Twest\_1 = Tupn\_2\ (T^m_{i-1,j,k})$ $Tupn\_1 = T^m_{i,j,k}$ $Tupn\_2 = T^m_{i+1,j,k}$ $Tngh\_1 = T^m_{i,j,k}$ $Tngh\_2 = T^m_{i+1,j,k}$ $Tsouth\_1 = T^m_{i,j,k+1}$ $Tsouth\_2 = T^m_{i+1,j,k+1}$		$Tngh\_1 = T^m_{i,j+1,k}$ $Tngh\_2 = T^m_{i+1,j+1,k}$	$Tngh\_1 = T^m_{i,j-1,k}$ $Tngh\_2 = T^m_{i+1,j-1,k}$	$Tngh\_1 = T^m_{i,j,k-1}$ $Tngh\_2 = T^m_{i+1,j,k-1}$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tsouth\_1 - Tupn\_1$ $dz2 = Tsouth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i-2,j,k): add contrib. L* $acc1 = acc1 + s1$ *Update node (i-1,j,k): add contrib. L* $acc2 = acc2 + s2$		*Update node (i-2,j,k): add contrib. N and S* $acc1 = acc1 + s1 + dz1$ *Update node (i-1,j,k): add contrib. N and S* $acc2 = acc2 + s2 + dz2$	*Update node (i-1,j,k): add contrib. E* $acc2t = acc2 - s1$ *Update node (i,j,k): contrib. E and W* $acc1 = s1 - s2$ *Update node (i+1,j,k): contrib. W* $acc2 = s2$	*Update node (i,j,k): add contrib. R* $acc1 = acc1 + s1$ *Update node (i+1,j,k): add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i-4,j,k): Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^{m+1}_{i-4,j,k})$ *Update node (i-3,j,k): add* $T^m_{i-3,j,k}$ $Tn\_2 = mult + Tunpprev\_2$ $(Tn\_2 = T^{m+1}_{i-3,j,k})$ *Update nodes (i-2,j,k) and (i-1,j,k):* $Tupnprev\_1 = Tupn\_1(T^m_{i-2,j,k})$ $Tupnprev\_2 = Tupn\_2(T^m_{i-1,j,k})$		*Update node (i-2,j,k): Load F0(i-2,j,k)* $F0 = F0(i-2,j,k)$	*Update node (i-2,j,k): mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i-1,j,k): load F0(i-1,j,k)* $F0 = F0(i-1,j,k)$	*Update node (i-2,j,k): add* $T^m_{i-2,j,k}$ $Tn\_2 = mult + Tupnprev\_1$ $(Tn\_2 = T^{m+1}_{i-2,j,k})$ *Update node (i-1,j,k): mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.13: Data flow on $PE0$ (I).

Clk	Step 6	Step 7	Step 8	Step 9	Step 10
Action on the memory banks	*Read* $T^m_{i+2,j,k}$, $T^m_{i+3,j,k}$ $T^m_{i+2,j,k+1}$, $T^m_{i+3,j,k+1}$	*Write* $T^{m+1}_{i-2,j,k}$ $T^{m+1}_{i-1,j,k}$	*Read* $T^m_{i+2,j+1,k}$, $T^m_{i+3,j+1,k}$	*Read* $T^m_{i+2,j-1,k}$, $T^m_{i+3,j-1,k}$	*Read* $T^m_{i+2,j,k-1}$, $T^m_{i+3,j,k-1}$
Load of Data in input registers	*New nodes to be updated: (i+2,j,k) and (i+3,j,k)* $Twest\_1 = Tupn\_2$ $(T^m_{i+1,j,k})$ $Tupn\_1 = T^m_{i+2,j,k}$ $Tupn\_2 = T^m_{i+3,j,k}$ $Tngh\_1 = T^m_{i+2,j,k}$ $Tngh\_2 = T^m_{i+3,j,k}$ $Tsouth\_1 = T^m_{i+2,j,k+1}$ $Tsouth\_2 = T^m_{i+3,j,k+1}$		$Tngh\_1 = T^m_{i+2,j+1,k}$ $Tngh\_2 = T^m_{i+3,j+1,k}$	$Tngh\_1 = T^m_{i+2,j-1,k}$ $Tngh\_2 = T^m_{i+3,j-1,k}$	$Tngh\_1 = T^m_{i+2,j,k-1}$ $Tngh\_2 = T^m_{i+3,j,k-1}$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tsouth\_1 - Tupn\_1$ $dz2 = Tsouth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i,j,k): add contrib. L* $acc1 = acc1 + s1$ *Update node (i+1,j,k): add contrib. L* $acc2 = acc2 + s2$		*Update node (i,j,k): add contrib. N and S* $acc1 = acc1 + s1 + dz1$ *Update node (i+1,j,k): add contrib. N and S* $acc2 = acc2 + s2 + dz2$	*Update node (i+1,j,k): add contrib. W* $acc2t = acc2 - s1$ *Update node (i+2,j,k): contrib. E and W* $acc1 = s1 - s2$ *Update node (i+3,j,k): contrib. E* $acc2 = s2$	*Update node (i+2,j,k): add contrib. R* $acc1 = acc1 + s1$ *Update node (i+3,j,k): add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i-2,j,k): Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^{m+1}_{i-2,j,k})$ *Update node (i-1,j,k): add* $T^m_{i-1,j,k}$ $Tn\_2 = mult + Tunpprev\_2$ $(Tn\_2 = T^{m+1}_{i-1,j,k})$ *Update nodes (i,j,k) and (i+1,j,k):* $Tupnprev\_1 = Tupn\_1(T^m_{i-2,j,k})$ $Tupnprev\_2 = Tupn\_2(T^m_{i-1,j,k})$		*Update node (i,j,k): Load F0(i,j,k)* $F0 = F0(i,j,k)$	*Update node (i,j,k): mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i+1,j,k): load F0(i+1,j,k)* $F0 = F0(i+1,j,k)$	*Update node (i,j,k): add* $T^m_{i,j,k}$ $Tn\_2 = mult + Tupnprev\_1$ $(Tn\_2 = T^{m+1}_{i,j,k})$ *Update node (i+1,j,k): mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.14: Data flow on $PE0$ (II).

Clk	Step 11	Step 12	Step 13	Step 14	Step 15
Action on the memory banks	*Read* $T^m_{i+4,j,k}$, $T^m_{i+5,j,k}$ $T^m_{i+4,j,k+1}$, $T^m_{i+5,j,k+1}$	*Write* $T^{m+1}_{i,j,k}$ $T^{m+1}_{i+1,j,k}$	*Read* $T^m_{i+4,j+1,k}$, $T^m_{i+5,j+1,k}$	*Read* $T^m_{i+4,j-1,k}$, $T^m_{i+5,j-1,k}$	*Read* $T^m_{i+4,j,k-1}$, $T^m_{i+5,j,k-1}$
Load of Data in input registers	*New nodes to be updated: (i+4,j,k) and (i+5,j,k)* $Twest\_1 = Tupn\_2\ (T^m_{i+3,j,k})$ $Tupn\_1 = T^m_{i+4,j,k}$ $Tupn\_2 = T^m_{i+5,j,k}$ $Tngh\_1 = T^m_{i+4,j,k}$ $Tngh\_2 = T^m_{i+5,j,k}$ $Tsouth\_1 = T^m_{i+4,j,k+1}$ $Tsouth\_2 = T^m_{i+5,j,k+1}$		$Tngh\_1 = T^m_{i+4,j+1,k}$ $Tngh\_2 = T^m_{i+5,j+1,k}$	$Tngh\_1 = T^m_{i+4,j-1,k}$ $Tngh\_2 = T^m_{i+5,j-1,k}$	$Tngh\_1 = T^m_{i+4,j,k-1}$ $Tngh\_2 = T^m_{i+5,j,k-1}$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tsouth\_1 - Tupn\_1$ $dz2 = Tsouth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i+2,j,k): add contrib. L* $acc1 = acc1 + s1$ *Update node (i+3,j,k): add contrib. L* $acc2 = acc2 + s2$		*Update node (i+2,j,k): add contrib. N and S* $acc1 = acc1 + s1 + dz1$ *Update node (i+3,j,k): add contrib. N and S* $acc2 = acc2 + s2 + dz2$	*Update node (i+3,j,k): add contrib. W* $acc2t = acc2 - s1$ *Update node (i+4,j,k): contrib. E and W* $acc1 = s1 - s2$ *Update node (i+5,j,k): contrib. E* $acc2 = s2$	*Update node (i+4,j,k): add contrib. R* $acc1 = acc1 + s1$ *Update node (i+5,j,k): add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i,j,k): Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^m_{i,j,k})$ *Update node (i+1,j,k): add* $T^m_{i+1,j,k}$ $(Tn\_2 = mult + Tupnprev\_2)$ $(Tn\_2 = T^{m+1}_{i+1,j,k})$ *Update nodes (i+2,j,k): and (i+3,j,k):* $Tupnprev\_1 = Tupn\_1(T^m_{i+2,j,k})$ $Tupnprev\_2 = Tupn\_2(T^m_{i+3,j,k})$		*Update node (i+2,j,k): Load F0(i+2,j,k)* $F0 = F0(i+2, j, k)$	*Update node (i+2,j,k): mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i+3,j,k): load F0(i+3,j,k)* $F0 = F0(i+3, j, k)$	*Update node (i+2,j,k) add* $T^m_{i+2,j,k}$ $Tn\_2 = mult + Tupnprev\_1$ $(Tn\_2 = T^{m+1}_{i+2,j,k})$ *Update node (i+3,j,k): mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.15: Data flow on $PE0$ (III).

Clk	Step 1	Step 2	Step 3	Step 4	Step 5
Action on the memory banks	*Read* $T^m_{i,j,k}$, $T^m_{i+1,j,k}$ $T^m_{i,j,k+1}$, $T^m_{i+1,j,k+1}$	*Write* $T^{m+1}_{i-4,j,k}$ $T^{m+1}_{i-3,j,k}$	*Read* $T^m_{i,j+1,k}$, $T^m_{i+1,j+1,k}$	*Read* $T^m_{i,j-1,k}$, $T^m_{i+1,j-1,k}$	*Read* $T^m_{i,j,k+1}$, $T^m_{i+1,j,k+1}$
Load of Data in input registers	*New nodes to be updated: (i,j,k) and (i+1,j,k)* $Twest\_1 = Tupn\_2$ $(T^m_{i-1,j,k})$ $Tupn\_1 = T^m_{i,j,k}$ $Tupn\_2 = T^m_{i+1,j,k}$ $Tngh\_1 = T^m_{i,j,k}$ $Tngh\_2 = T^m_{i+1,j,k}$ $Tsouth\_1 = T^m_{i,j,k+1}$ $Tsouth\_2 = T^m_{i+1,j,k+1}$		$Tngh\_1 = T^m_{i,j+1,k}$ $Tngh\_2 = T^m_{i+1,j+1,k}$	$Tngh\_1 = T^m_{i,j-1,k}$ $Tngh\_2 = T^m_{i+1,j-1,k}$	$Tngh\_1 = T^m_{i,j,k+1}$ $Tngh\_2 = T^m_{i+1,j,k+1}$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tsouth\_1 - Tupn\_1$ $dz2 = Tsouth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i-2,j,k):* *add contrib. L* $acc1 = acc1 + s1$ *Update node (i-1,j,k):* *add contrib. L* $acc2 = acc2 + s2$		*Update node (i-2,j,k):* *add contrib. S* $acc1 = acc1 + s1 + dz1$ *Update node (i-1,j,k):* *add contrib. S* $acc2 = acc2 + s2 + dz2$	*Update node (i-1,j,k):* *add contrib. E* $acc2t = acc2 - s1$ *Update node (i,j,k):* *contrib. E and W* $acc1 = s1 - s2$ *Update node (i+1,j,k):* *contrib. W* $acc2 = s2$	*Update node (i,j,k):* *add contrib. R* $acc1 = acc1 + s1$ *Update node (i+1,j,k):* *add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i-4,j,k):* *Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^{m+1}_{i-4,j,k})$ *Update node (i-3,j,k):* *add boundary_2* $Tn\_2 = mult + boundary\_2$ $(Tn\_2 = T^{m+1}_{i-3,j,k})$		*Update node (i-2,j,k):* *Load F0(i-2,j,k)* $F0 = F0(i-2, j, k)$	*Update node (i-2,j,k):* *mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i-1,j,k):* *load F0(i-1,j,k)* $F0 = F0(i-1, j, k)$	*Update node (i-2,j,k):* *add boundary_1* $Tn\_2 = mult + boundary\_1$ $(Tn\_2 = T^{m+1}_{i-2,j,k})$ *Update node (i-1,j,k):* *mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.16: Data flow on $PE0$ processing surface nodes (I).

Clk	Step 6	Step 7	Step 8	Step 9	Step 10
Action on the memory banks	*Read* $T^m_{i+2,j,k}$, $T^m_{i+3,j,k}$ $T^m_{i+2,j,k+1}$, $T^m_{i+3,j,k+1}$	*Write* $T^{m+1}_{i-2,j,k}$ $T^{m+1}_{i-1,j,k}$	*Read* $T^m_{i+2,j+1,k}$, $T^m_{i+3,j+1,k}$	*Read* $T^m_{i+2,j-1,k}$, $T^m_{i+3,j-1,k}$	*Read* $T^m_{i+2,j,k+1}$, $T^m_{i+3,j,k+1}$
Load of Data in input registers	*New nodes to be updated: (i+2,j,k) and (i+3,j,k)* $Twest\_1 = Tupn\_2$ $(T^m_{i+1,j,k})$ $Tupn\_1 = T^m_{i+2,j,k}$ $Tupn\_2 = T^m_{i+3,j,k}$ $Tngh\_1 = T^m_{i+2,j,k}$ $Tngh\_2 = T^m_{i+3,j,k}$ $Tsouth\_1 = T^m_{i+2,j,k+1}$ $Tsouth\_2 = T^m_{i+3,j,k+1}$		$Tngh\_1 = T^m_{i+2,j+1,k}$ $Tngh\_2 = T^m_{i+3,j+1,k}$	$Tngh\_1 = T^m_{i+2,j-1,k}$ $Tngh\_2 = T^m_{i+3,j-1,k}$	$Tngh\_1 = T^m_{i+2,j,k+1}$ $Tngh\_2 = T^m_{i+3,j,k+1}$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tsouth\_1 - Tupn\_1$ $dz2 = Tsouth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i,j,k): add contrib. L* $acc1 = acc1 + s1$ *Update node (i+1,j,k): add contrib. L* $acc2 = acc2 + s2$		*Update node (i,j,k): add contrib. S* $acc1 = acc1 + s1 + dz1$ *Update node (i+1,j,k): add contrib. S* $acc2 = acc2 + s2 + dz2$	*Update node (i+1,j,k): add contrib. W* $acc2t = acc2 - s1$ *Update node (i+2,j,k): contrib. E and W* $acc1 = s1 - s2$ *Update node (i+3,j,k): contrib. E* $acc2 = s2$	*Update node (i+2,j,k): add contrib. R* $acc1 = acc1 + s1$ *Update node (i+3,j,k): add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i-2,j,k): Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^{m+1}_{i-2,j,k})$ *Update node (i-1,j,k): add boundary_2* $Tn\_2 = mult + boundary\_2$ $(Tn\_2 = T^{m+1}_{i-1,j,k})$		*Update node (i,j,k): Load F0(i,j,k)* $F0 = F0(i,j,k)$	*Update node (i,j,k): mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i+1,j,k): load F0(i+1,j,k)* $F0 = F0(i+1,j,k)$	*Update node (i,j,k): add boundary_1* $Tn\_2 = mult + boundary\_1$ $(Tn\_2 = T^{m+1}_{i,j,k})$ *Update node (i+1,j,k): mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.17: Data flow on $PE0$ processing surface nodes (II).

Clk	Step 11	Step 12	Step 13	Step 14	Step 15
Action on the memory banks	*Read* $T^m_{i+4,j,k}$, $T^m_{i+5,j,k}$ $T^m_{i+4,j,k-1}$, $T^m_{i+5,j,k-1}$	*Write* $T^{m+1}_{i,j,k}$ $T^{m+1}_{i+1,j,k}$	*Read* $T^m_{i+4,j+1,k}$, $T^m_{i+5,j+1,k}$	*Read* $T^m_{i+4,j-1,k}$, $T^m_{i+5,j-1,k}$	*Read* $T^m_{i+4,j,k+1}$, $T^m_{i+5,j,k+1}$
Load of Data in input registers	*New nodes to be updated: (i+4,j,k) and (i+5,j,k)* $Twest\_1 = Tupn\_2$ $(T^m_{i+3,j,k})$ $Tupn\_1 = T^m_{i+4,j,k}$ $Tupn\_2 = T^m_{i+5,j,k}$ $Tngh\_1 = T^m_{i+4,j,k}$ $Tngh\_2 = T^m_{i+5,j,k}$ $Tsouth\_1 = T^m_{i+4,j,k+1}$ $Tsouth\_2 = T^m_{i+5,j,k+1}$		$Tngh\_1 = T^m_{i+4,j+1,k}$ $Tngh\_2 = T^m_{i+5,j+1,k}$	$Tngh\_1 = T^m_{i+4,j-1,k}$ $Tngh\_2 = T^m_{i+5,j-1,k}$	$Tngh\_1 = T^m_{i+4,j,k+1}$ $Tngh\_2 = T^m_{i+5,j,k+1}$
Operations in SUB block	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$		$s1 = Twest\_1 - Tupn\_1$ $s2 = Tupn\_1 - Tupn\_2$ $dz1 = Tsouth\_1 - Tupn\_1$ $dz2 = Tsouth\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$	$s1 = Tngh\_1 - Tupn\_1$ $s2 = Tngh\_2 - Tupn\_2$
Operations in ACC block	*Update node (i+2,j,k): add contrib. L* $acc1 = acc1 + s1$ *Update node (i+3,j,k): add contrib. L* $acc2 = acc2 + s2$		*Update node (i+2,j,k): add contrib. N and S* $acc1 = acc1 + s1 + dz1$ *Update node (i+3,j,k): add contrib. N and S* $acc2 = acc2 + s2 + dz2$	*Update node (i+3,j,k): add contrib. W* $acc2t = acc2 - s1$ *Update node (i+4,j,k): contrib. E and W* $acc1 = s1 - s2$ *Update node (i+5,j,k): contrib. E* $acc2 = s2$	*Update node (i+4,j,k): add contrib. R* $acc1 = acc1 + s1$ *Update node (i+5,j,k): add contrib. R* $acc2 = acc2 + s2$
Operations in Mult-Output block	*Update node (i,j,k): Transfer Tn_2 to Tn_1* $Tn\_1 = Tn\_2$ $(Tn\_1 = T^m_{i,j,k})$ *Update node (i+1,j,k): add* $T^m_{i+1,j,k}$ $Tn\_2 = mult + boundary\_2$ $(Tn\_2 = T^{m+1}_{i+1,j,k})$		*Update node (i+2,j,k): Load F0(i+2,j,k)* $F0 = F0(i+2, j, k)$	*Update node (i+2,j,k): mult. contrib. neig. by F0* $mult = acc1 * F0$ *Update node (i+3,j,k): load F0(i+3,j,k)* $F0 = F0(i+3, j, k)$	*Update node (i+2,j,k) add boundary_1* $Tn\_2 = mult + boundary\_1$ $(Tn\_2 = T^{m+1}_{i+2,j,k})$ *Update node (i+3,j,k): mult. contrib. neig. by F0* $mult = ac2t * F0$

Table 3.18: Data flow on $PE0$ processing surface nodes (III).

	Step -2	Step -1	Step1	Step2	Step3	Step4
add_11	$2F_{01}H_1 + mult\_b1$	$mult\_b1 + T_{i-2,j,1}$	$add\_11 + mult\_b1$		$T_{air} - T_{i,j,1}$	$2F_{01}H_1 + mult\_b1$
add_12	add_11					add_11
add_21	$2F_{02}H_2 + mult\_b2$	$mult\_b2 + T_{i-1,j,1}$	$add\_21 + mult\_b2$		$T_{air} - T_{i+1,j,1}$	$2F_{02}H_2 + mult\_b2$
add_22	add_21					add_21
$mult\_b1$	$2F_{01}A_1S_1 \cdot q_{sun}$	$add\_11 \cdot add\_12$			$T_{air}^3 \cdot 8F_{01}R_1$	$2F_{01}A_1S_1 \cdot q_{sun}$
$mult\_b2$	$2F_{02}A_2S_2 \cdot q_{sun}$	$add\_21 \cdot add\_22$			$T_{air}^3 \cdot 8F_{02}R_2$	$2F_{02}A_2S_2 \cdot q_{sun}$
$boundary\_1$					add_11	
$boundary\_2$					add_21	

Table 3.19: Data flow on adders/subtractors and multipliers of the *Boundary* block in $PE0$ (I).

	Step5	Step6	Step7	Step8	Step9	Step10
add_11	$mult\_b1 + T_{i,j,1}$	$add\_11 + mult\_b1$		$T_{air} - T_{i+2,j,1}$	$2F_{01}H_1 + mult\_b1$	$mult\_b1 + T_{i+2,j,1}$
add_12					add_11	
add_21	$mult\_b2 + T_{i+1,j,1}$	$add\_12 + mult\_b2$		$T_{air} - T_{i+3,j,1}$	$2F_{02}H_2 + mult\_b2$	$mult\_b2 + T_{i+3,j,1}$
add_22					add_21	
$mult\_b1$	$add\_11 \cdot add\_12$			$T_{air}^3 \cdot 8F_{01}R_1$	$2F_{01}A_1S_1 \cdot q_{sun}$	$add\_11 \cdot add\_12$
$mult\_b2$	$add\_21 \cdot add\_22$			$T_{air}^3 \cdot 8F_{02}R_2$	$2F_{02}A_2S_2 \cdot q_{sun}$	$add\_21 \cdot add\_22$
$boundary\_1$				add_11		
$boundary\_2$				add_21		

Table 3.20: Data flow on adders/subtractors and multipliers of the *Boundary* block in $PE0$ (II).

Table 3.21: FPGA utilization resources.

Resource	Utilization
Slices	7201 (21%)
18x18 Multipliers	10 (7%)
RAMB16s	138 (96%)
External IOBs	573 (54%)

3.4.6 Characteristics of the implementation

In this section we will summarize the main aspects related to the FPGA implementation. As was said the system was designed using VHDL to design the PE and all the control circuitry, whereas Handel-C was used to design the interface of the $HOST$ with the memory banks and the FPGA. A clock of 50MHz was achieved, which is limited by the multiplicators.

As mentioned previously, the FPGA utilization is not very high and there is a big number of available slices, see Table 3.21. We could think of introducing more PEs if more memory banks were available, what would allow to process more nodes in parallel. Taking into account that each PEk (k=1...6) consumes 600 slices (PE0 consumes 950 slices) and assuming a 10% of over cost to do the routing we have estimated that at least 40 new PEs could be introduced in the system due to the limitations in the number of slices. However, the number of PEs is also limited by the $IOBs$. Since each PE consumes 59 $IOBs$, there are still 487 $IOBs$ available, thus 8 new PEs could be added to the system by $IOBs$ limitations. Thus, if 8 memory banks were introduced in the system, 8 new PEs could be added, which would increase the speedup. Thus, this system is based in a scalable architecture which has the limits of the $IOBs$ with the current FPGA.

3.5 Summary

In this chapter we have introduced the heat equation hardware solver. Firstly we have introduced the hardware used to carry out the implementation. Then a detailed description of the architecture and all implementations issues was done, emphasizing the data flow. Finally, we have proofed the scalability of the architecture and how changing the FPGA a

system with more computing power could be achieved.

Chapter 4

Results

4.1 Introduction

As was said in Chapter 1, a technique to detect buried landmines or objects was proposed in (López, 2003; López et al., 2004). In this technique, based on IRT, the detection process can be done due to the thermal contrast soil-mine during the sunrise (around 4-6 ℃). However, the application of this technique is a very-long time-consuming process in ordinary computers. The main objective of our work was to reduce the computing time of this technique to allow an on-field application. To overcome this drawback we have worked in two ways: a FPGA implementation which executes the thermal model and the use of non-uniform grids to reduce the number of layer of the thermal model. We will evaluate the FPGA implementation (integer arithmetic), both the errors introduced due to the use of integer arithmetic and the achieved speedup. We will also study the use of non-uniform grids to reduce the number of layers of the thermal model. We will show that the errors introduced in both cases are much smaller than the thermal contrast soil-mine. To this aim we will compare the data from two experiments (indoor and outdoor) which were used in (López, 2003; López et al., 2004) to validate the detection process.

4.2 Experimental setups

In this section we will describe the most important aspect of the two experimental setups used in the validation of the FD-TD hardware solver and on the use of non-uniform grids.

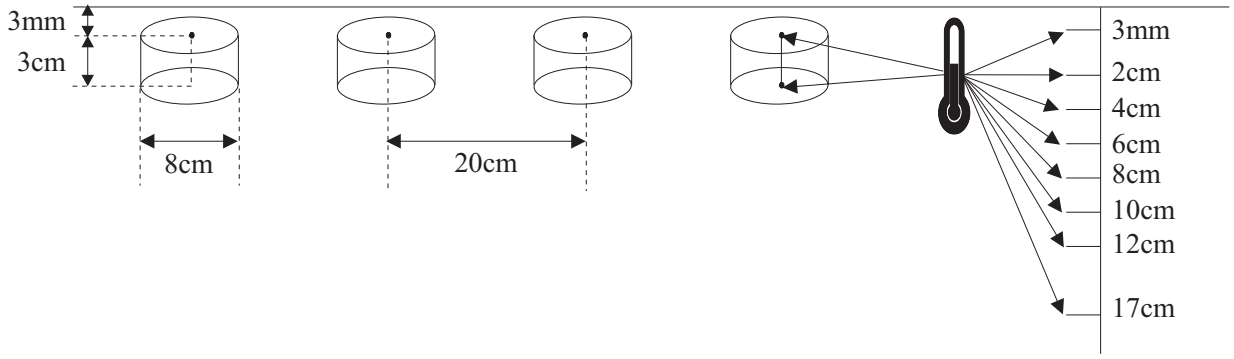


Figure 4.1: Experimental setup used in indoor scenario: placement of the test mines and the temperature probes in the soil.

We will do a brief introduction of the experimental setups.

4.2.1 Indoor scenario

This scenario corresponds to the indoor test facilities of the Defense research Establishment (FOA) of the Swedish Armed Forces. Al full description of the setup can be found in (Maksymonko et al., 1995; Sjokvist, 1999).

The setup consists of a dry sand box of dimensions 1m × 1m × 0.4m for length, width and depth respectively. A solar panel, which produces an insolation distribution similar to that of the sun, produces the artificial heating of the soil. The IR camera used to obtain the images was an AGEMA Thermovision 900, which gives a resolution of 272 × 136 pixels. The IR cameras were mounted in the center of the solar panel, 2.1 m above the sand surface. In this setup a system of Pt-100 thermometers probes was used to record the temperature at different depths. The experimental setup can be seen in Fig. 4.1. Four test mines were shallowly laid, all cylindrical, with a diameter of 8 cm and 3 cm thick. The distances between the center of the test mines was 20 cm. For recording the temperatures of the mines, the probes were attached to the center of the mines on the upper side as well as on the underside. For recording the in-depth temperature of the soil the probes were placed far away from the mines in order to measure the unperturbed behavior. The values of the thermal properties of the sand and the surrogated mine considered in this scenario are summarized in Table 4.1

Table 4.1: Indoor scenario: Physical characteristics of the sandy soil and the surrogated mines.

	Dry sand	Antipersonnel mine
ρ(kg/m^3)	1650	1350
c_p(J/kgK)	710	1120
k(W/mK)	0.75	0.4

Table 4.2: Outdoor scenario: Physical characteristics of the sandy soil.

Dry matter density (kg/m^3)	1525
Iron content (g/kg)	1.9
Organic matter content	$< 1\%$
Luttun content	$< 2\%$
Ground water level	120 cm

4.2.2 Outdoor scenario

This scenario corresponds to the outdoor test facilities of the TNO Physics and Electronics Laboratory (TNO-FEL), in the Netherlands. A detailed description of the setup can be found in (de Jong et al., 1999; de Jong and Sjokvist, 2002). The images that we will use are of a sand lane with dimensions 10m×3m×1.5m. An image of the TNO Physics and Electronics Laboratory sand lane can be seen in Fig. 4.2. In Table 4.2 the characteristics of the sand lane are shown. In this lane 7 types of AP mines and 3 types of AT mines are present. Fig. 4.3 shows the ground truth of the southern part sand lane. The characteristics of the tests mines are shown in Table 4.3. Additionally, in the sand lane there are non-mine objects objects, which are summarized in Table 4.4. All mines are surrogated and most of then has been built at TNO-FEL, (de Jong et al., 1999). In all these mines the explosive has been replaced by a material with the same relevant properties as the real explosive. The most common material to simulate the properties of the explosives, such as TNT, composition B-3 and Tetryl, is the silicone rubber RTV3110, (Corning, 1981). The thermal properties of the silicone rubber RTV3110 are given in Table 4.5.

Figure 4.2: Photo of the sand lane.

Camera specifications

During the obtention of the images several cameras were used to obtain images in different spectral ranges. However, we are going to consider only the IR images. The camera employed and the experimental setup used to take the images are shown in Fig. 4.4. The characteristics of the camera, a Quantum Well IR Photodetector (QWIP), ThermaCAM SC3000, are shown in Table 4.6, (Systems, 2005).

The spectral band of the camera is in the 8-9 μm range, which corresponds to the spectral window in which the attenuation caused by the atmosphere of the earth is minor. For out objective the most important parameter of the IR camera is the MRTD (Minimum Resolvable Temperature), that is defined as the differential temperature of a target that can be resolvable by the sensor. As can be seen in Table 4.6, the MRTD of the camera is 0.03 K, which is far beyond of our requirements since the thermal contrast between mine as soil is of several degrees.

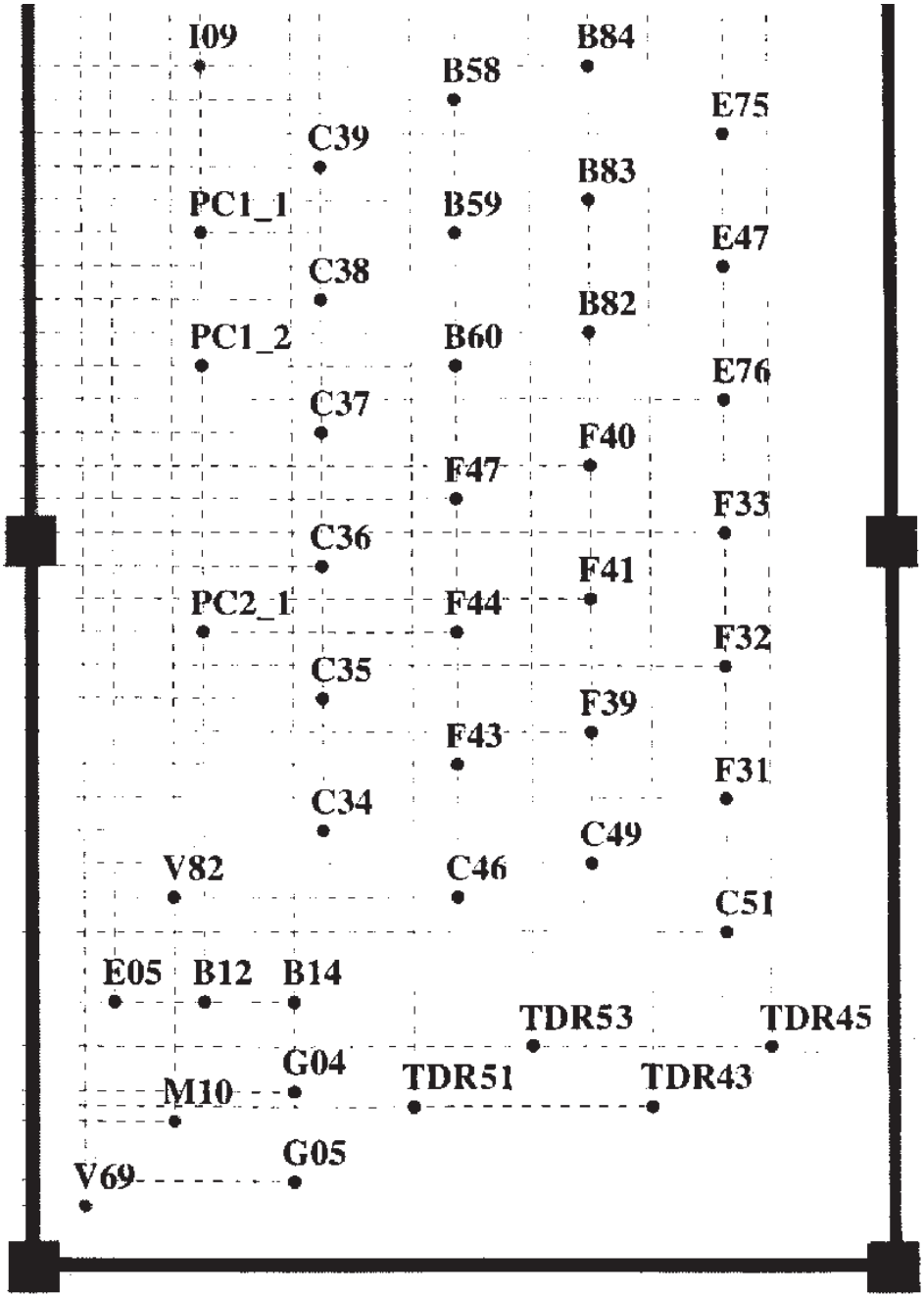


Figure 4.3: Ground truth of the sand lane - courtesy of TNO-FEL Laboratory.

Table 4.3: Description of the AP and AT mines present in the sand lane.

Code	Mine Type	Radius(mm)	Height(mm)	Weight(g)	Case	Metal
B**	AP	31	53	90	plastic	non
C**	AP	58.2	48	669	PVC	medium
E**	AP	31	53	86	plastic	non
F**	AP	27.5	42	82	ABS	low
G**	AP	29.5	18	60	PE	medium
I**	AT	127	134	8500	metal	high
L**	AT	157.5	150	5900	PVC	low
M**	AT	150	115	7000	no casing	non

Table 4.4: Outdoor scenario: Description of non-mine objects present in the sand lane.

Code	Description
PC**	Undefined test object
V81, V82	Test object (metal disk)
V69, V74	Thermocouple stick
TDR**	TDR measurement probes

4.3 Practical considerations

In this section we sill describe those aspects that must be taken into account before using the model to process IR images acquired in real scenarios: resolution, calibration, initialization, meteorological data (air temperature, sun radiation and wind speed), moisture content and thermal properties.

4.3.1 Resolution

Some considerations related to the size and resolution of the IR images must be made before using the model. The size of the images depends on the specific sensor used during the acquisition process and it has a strong influence in the field of view (FOV) and in the

(a) Skylift with camera above the sand

(b) Camera mounting setup

Figure 4.4: Setup to obtain the IR images in the outdoor scenario.

resolution of the system in terms of number of pixels per centimeter. The FOV varies not only as a function of the height at which the camera is placed, but also of its tilting angle. There is an upper limit in the height of the camera. The bigger the height, the lesser number of pixels per centimeter of soil, which deteriorates the resolution and quality of the images. The critical point is given by the size of the minimum object that has to be detected, to which a minimum of 4-5 pixels must be associated. This fixes the Δx and Δy parameters of the FD-TD model.

This is related with the choice of the spatial discretization parameters of the thermal model Δx and Δy, which give the accuracy of the thermal model and have influence on the stability of the system. If each pixel of the image is mapped into a node of the grid, then the Δx and Δy are equal to R_H (horizontal resolution) and R_V (vertical resolution).

Table 4.5: Thermal properties of the silicone rubber RTV3110.

ρ (kg/m^3)	1170
c_p (J/kgK)	1500
k (W/mK)	0.2

Table 4.6: QWIP camera specifications.

Spectral range	8-9 μm
Detector	GaAs, Quantum Well
Cooling	Strirling cooled to 70 K
Field of view	2'×15 degrees
Frame rate	50/60 Hz
Picture frame	320 × 240
MRTD	0.03 K
Temperature range	-20℃- 1500℃

The Δz parameter must be also chosen carefully. As was said, when using IRT for landmine detection the maximum depth of burial at which mines can be detected is around 10-15 cm. On the other hand, it can be assumed that the exact distribution of temperatures at deep depths does not influence the surface effect. For this reason, a sufficiently deep volume must be considered during the simulations to fulfill the deep ground condition introduced in the boundary conditions of the problem:

$$T(x, y, z \to \infty, t) = T_\infty \tag{4.1}$$

that defines the depth at which the temperature remains unaltered by the phenomena that are taking place at the soil-air interface, this depth is around 40-50 cm. As happens with Δx and Δy, a Δz parameter must be chosen so that the object that has to be detected have associated several layers.

4.3.2 Calibration

The thermal model works with real images acquired with commercial IR cameras. The grayscale values of these images correspond to an apparent temperature on the surface of the soil. Most cameras perform the conversion to temperature images automatically for each acquired frame. This calibration differs from frame to frame due to internal adjustments of the camera, especially for acquisition experiments that take place during a long time interval.

The temperature map obtained with the IR camera is expressed in apparent temper-

atures. This is a mix of reflected and emitted radiation as seen by the sensor, but not the real temperature. The correspondence between apparent and real temperatures is not straightforward, being necessary to introduce a radiometric model describing the spatial and spectral characteristics of the radiation transmitted from the soil surface to the IR detector. Such a model requires the knowledge of the spectral behavior of the source and the soil, as well as the wavelength dependence from the detectors, optics, filters and the atmosphere, (Sendur and Baertlein, 2000). However, dynamic IR thermography relies on the analysis of the target-background temperature difference, rather than in the absolute values and therefore working with either real or apparent temperatures is equivalent.

4.3.3 Initialization of the thermal model

In order to initialize the model, the initial distribution of temperatures in the soil is needed. The acquired IR image will be associated to the first layer of the 3D grid, being the initial distribution of temperatures at the surface. The main problem is how to perform the initialization of the temperatures inside the soil volume. Some thermocouples can be placed in minefields at different depths to obtain the initial distribution of temperatures, however this is not practical for on-field applications. Some authors have estimated the distribution of temperatures inside the soil as, (Pregowski et al., 2000):

$$T\left(z,t\right) = T_0 - r\ z + T_a e^{-z/l} sin\left(\omega t - \frac{z}{l}\right) \tag{4.2}$$

where T_0[℃] is the mean temperature of the soil surface, T_a[℃] is the amplitude of the temperature variation at the soil surface, r[℃/cm] is the coefficient of mean temperatures changes with depth, l is the attenuation depth at which the temperature decreases e times with respect to surface temperature and $\omega = 2\pi T^{-1}$ is the angular frequency, where T is the period. The estimate of the r and l coefficients is equivalent to characterizing the soil signature, but it is of not use when buried objects have been detected.

4.3.4 Meteorological data

In this section we will discuss different aspects related to the meteorological data used in the thermal model, i.e., the air temperature, the irradiance of the sun and wind speed.

During the experiments, the air temperature can be estimated, (England, 1990), obtained from the IR images or measured using a meteorological station. As the variation

range of air temperatures is 10-20℃ either of these approaches are valid. However, the simplest choice is to have a meteorological station which give us the real values of the air temperature.

Under conditions of natural heating, it is necessary to measure or estimate the irradiance from the sun. The simplest way of obtaining this value is to measure it with an appropriate sensor. However, if this value cannot be measured during the experiments, an estimate is possible using this expression:

$$q_{sun}(t) = S_0(1 - al)(1 - C)H(t) \tag{4.3}$$

where S_0 is the solar constant, al is the ground albedo, C is a factor that accounts for the solar flux reduction due to the cloud cover and $H(t)$ is the local insolation function, (Watson, 1973; Sendur and Baertlein, 2000).

Finally, the wind speed can be assumed to be constant during the experiments. However, its value can be measured and its variations will affect to the convective heat transfer coefficient.

4.3.5 Moisture content

In the thermal model used in this work we assumed that the moisture content in the soil is constant. For this reason, no dependence between the physical properties of the soil, that will keep a constant value, and the moisture content were taken into account.

4.3.6 Thermal properties

The thermal behavior of the soil is given by its thermal properties. Thus, to perform accurate simulations we must know with precision the following properties:

- Density.
- Specific heat.
- Thermal conductivity.

The exact value of these properties depends on the moisture content that can vary according to climatological conditions.

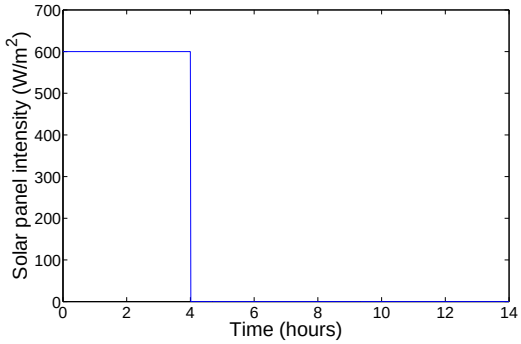
Thus, taking into account all these considerations, we see that the real problem is very complicated. However, we have assumed that both soil and buried objects can be modeled as isotropic objects. We will also assume that the moisture content of the soil during the analysis can be neglected. As the detection technique uses a temporal window of 1 hour the effects of mass transfer can be also neglected, and it can be assumed that the thermal properties of the soil and buried objects remain unperturbed during the analysis.

4.4 Thermal model test

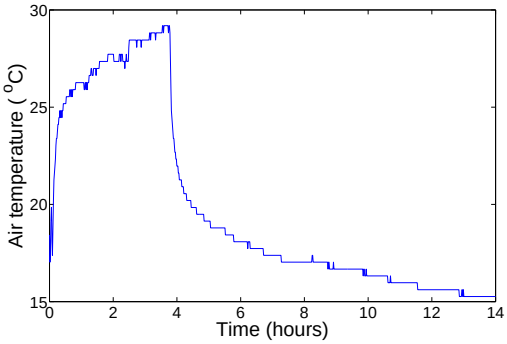
In this section we will validate the modifications proposed to the original thermal model comparing the results obtained through the FPGA (integer arithmetic) and by means of PC simulations (double precision, C++). We will also study the effects of using non-uniform grids. To this aim we have used the data from two experiments with different heat conditions.

4.4.1 Full radiation test

In this indoor experiment a solar panel, with a constant intensity of 600 W/m^2, radiates a piece of soil for four hours and after this time it is turned off, see Fig. 4.5(a). The evolution of the air temperature was measured with a temperature probe located just above the sand surface, see Fig. 4.5(b). In the experimental setup some probes were located at different depths on the sand; these values were used and extrapolated to obtain the initial temperature values. In this test, the considered piece of soil is far from the mines and, thus, we can considered an uniform volume of soil ($\alpha_{soil} = 6.4 \cdot 10^{-7}$ m^2/s). As it is an indoor experiment the velocity of the air can be controlled and a value of $h = 5$ W/m^{2}K was used. The value of the solar radiation is set by the solar panel and the air temperature was measured with a probe. This test was considered in Chapter 3 to study the wordlength influence of the temperature and the remaining parameters in the precision of the computations. Now we will analyze the effects of the implementation on the FPGA using integer arithmetic and the use of non-uniforms grids to reduce the number of nodes that must be computed.



(a) Full radiation test: radiation with the solar panel



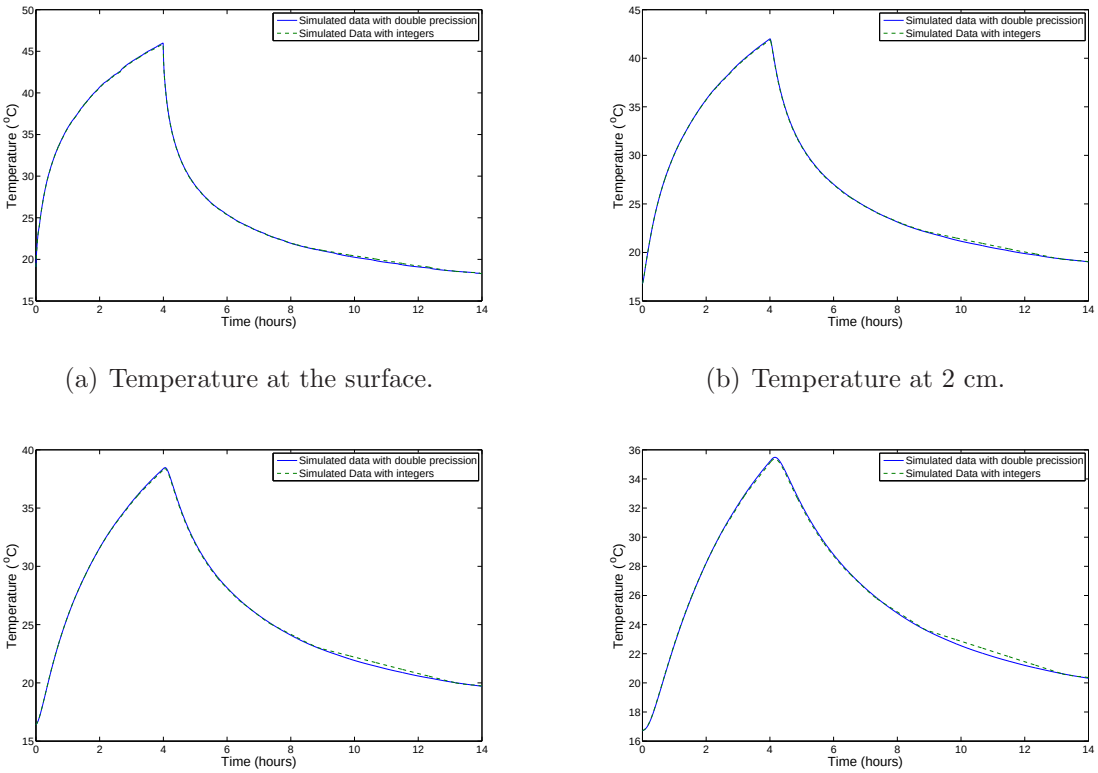
(b) Air temperature

Figure 4.5: Full radiation test: Values of the boundary conditions.

Full radiation test: Integer Arithmetic

In this example only soil points are involved on the simulation and a volume of $10 \times 10 \times 40$ nodes was considered. The spatial discretization was chosen to be $\Delta x = \Delta y = \Delta z = 1$ cm and the temporal discretization $\Delta t = 10s$. Given the value $\alpha_{soil} = 6.4 \cdot 10^{-7}$ m^2/s, a value of $F_0 = 0.06$ is obtained, which satisfies the stability criteria. The evolution of the temperatures was studied for 14 hours. The temperatures obtained simulating the system with double precision are compared with those obtained using integer arithmetic on the FPGA, the results are shown in Figs 4.6(a)-(d) for sand surface, at 2 cm, 4 cm and 6 cm depth, respectively. As can be seen the agreement between both simulations is very good. For the surface layer the agreement of the data is excellent through all the process. The maximum absolute differences between both simulations are collected in Table. 4.7. It can be noted that these errors are much more smaller than the thermal contrast soil-mine (5-6 ℃) and therefore the integer arithmetic is suitable for the detection algorithm. It can be seen in Fig. 4.6 that while the soil is being radiated with the solar panel the error in the system is lower than at the end of the simulation. In fact, the maximum error at the surface while the soil is being radiated and four hours after the solar panel was turned off is only 0.1 ℃. As the detection algorithm works with images at the sunrise (heating process) and lasts for 1 hour, the errors introduced will be expected to be in this range.

In Table 4.8 the different actions associated with the processing of the previous experiment in the FPGA implementation are summarized. It can be noted that the time required to perform the data conversion from float to integer is negligible when compared to the



(a) Temperature at the surface.

(b) Temperature at 2 cm.

(c) Temperature at 4 cm.

(d) Temperature at 6 cm.

Figure 4.6: Full radiation test: Comparison between the simulated data obtained using double precision and those obtained using integer arithmetic.

total processing time and so is the initialization time of the FPGA. It can also be seen that getting intermediate results from the memory banks is not a critical issue. The total speedup factor of the FPGA design with respect to a Pentium IV 3 GHZ single precision implementation (C++) for this experiment is 34, which demonstrates the validity of the approach as an efficient heat equation solver and hardware accelerator. It is important to note that in this test the computations were stopped every 20 minutes to get intermediate results, which have increased in 0.84 s in the total processing time.

Full radiation test: Non-uniform grid

In this section we will study the effects of using a non-uniform grid space between layers in the z axis. All the simulations with non-uniform grids were carried out using MATLAB (double precision). From now on we will only consider the temperatures at the surface

Table 4.7: Full radiation test: maximum differences between simulation performed in double precision and integer arithmetic.

Depth	Maximum difference between simulations (℃)
Surface	0.17
2 cm	0.23
4 cm	0.31
6 cm	0.35

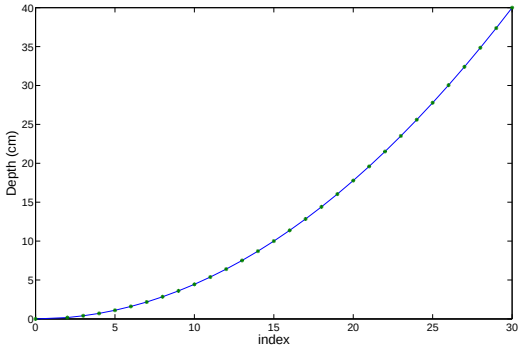
Table 4.8: Times for different FPGA events.

Event	Time (s)
Data integer conversion	0.052
Initialization of the FPGA	0.15
Processing time (VHDL)	50
Upload data from the memory banks	0.02

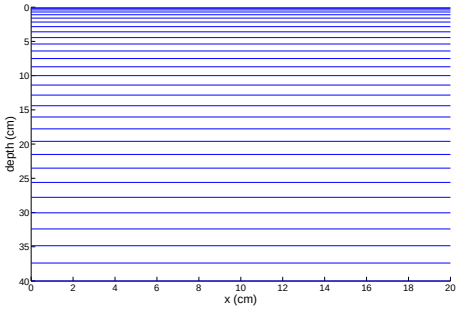
because the detection algorithm only involves surface temperatures. As was mentioned in the previous chapter as the depth of burial is increased the effect of the heating on soil' temperature decreases and the presence of the mine does not produce any perturbation on the soil thermal behavior. Thus, for a location deeper from 15-20 cm it is useless to use a high precision in the spatial domain because the thermal contrast decreases. As was pointed in Chapter 2, the grid variation depends on the considered problem. This first function that we have used to vary the space between layers was:

$$z(i) = L * \left(\frac{i}{N}\right)^2 \tag{4.4}$$

where L is the depth that we want to achieve and N is the number of points that we want to use to cover such a depth. This function was adapted from the scheme proposed in (de Rivas, 1972). Applying this discretization on this example we have obtain the z points and the grid shown in Fig. 4.7. It can be noted how at lower depths the space between z-layers is smaller than at higher depths. Using this scheme we obtain higher resolution at lower depths, where the IRT technique is effective. However, this discretization has an important drawback in terms of time, this is due to the fact that the Δz discretization step in the first layers is very low and to achieve a valid solution the Δt discretization step



(a) Points used on the discretization on the z direction.



(b) Transversal view of the grid

Figure 4.7: Points used in the z direction to perform the discretization and transversal view of the grid using Eq. (4.4).

must be smaller than using the uniform grid with $\Delta z = 1$ cm. Thus, although the number of layers is reduced, from 40 to 30, the number of iterations that must be performed is much more higher than using uniform grid and therefore this grid is useless to reduce the computing time. We will illustrate this issue with a numerical example. Let's consider this example, where the uniform grid has $10 \times 10 \times 40$ nodes; as $L = 40$, if we take $N = 30$ a reduction of 25 % in the number of nodes is achieved. Now we will analyze the effects of such a discretization in the stability condition. Using the uniform grid with $\Delta t = 10$ s and $\Delta z = 1$ cm, $F_0 = 0.064$, the stability condition:

$$F_0 = \alpha \frac{\Delta t}{(\Delta x)^2} \leq \frac{1}{6} \tag{4.5}$$

is satisfied. Using a non-uniform grid the stability condition can be written as:

$$F_0 = \frac{\alpha \Delta t}{(\Delta d)^2} \tag{4.6}$$

where $\Delta d = min(\Delta z_1, ...\Delta z_n, \Delta x)$. Using this non-uniform grid, maintaining $\Delta t = 10\,s$ and taking into account that the first (the smallest value) Δz value obtained with this scheme is 0.04 cm, a value of $F_0 = 40$ is obtained, which does not satisfies the stability condition given by Eq. (4.6). Thus, the temporal discretization step must be changed to a value of $\Delta t = 0.04\,s$ to satisfy Eq. (4.6). However, the number of iterations to complete the same simulation time is considerably increased. In Table 4.9 a comparison between the simulation using uniform and non-uniform grids is done, where we have considered a simulation time of 14 hours. It can be seen how the product number of iteration by number

Table 4.9: Comparison of the simulations using uniform and non-uniform grid given by Eq. (4.4).

	Uniform grid	Non-uniform grid
$\Delta t\,(s)$	10	0.04
$\Delta z\,(cm)$	1cm	0.04 - 3
Iterations	5040	1260000
Iterations×Number of grid points	2016×10^4	378×10^7

of grid points, which is proportional to the number of operations that must be done, is three order of magnitude higher using the non-uniform grid.

Taking into account the limitation that this scheme presents, we have used a different approach. In landmine detection by means of dynamic thermography the maximum depth of burial at which mines can be detected is around 10-15 cm, (López, 2003); thus we will used an uniform grid until 20 cm and for higher depths the grid space is increased. A first approach is to increase the Δz value at 20 cm and then maintain it constant. Fig. 4.8 shows the simulation using different Δz parameters from 20 cm. It can be noted that as bigger the Δz discretization step is bigger the error is. The maximum differences between the temperatures using the uniform grid and those used these non-uniform grids are shown in Table 4.10. In this table is also collected the reduction percentage in the number of layers, which is translated in a reduction in the computing time; a Trade-off (TO) factor defined as:

$$TO = \frac{Maximum\ \ error}{Reduction} \tag{4.7}$$

where *Reduction* is the reduction percentage in the number of layers that must be computed.

We have used another non-uniform grid; in this case the Δz parameter is maintained constant (1 cm) until 20 cm and it varies in the form:

$$\Delta z_k = \Delta z_{k-1} + \beta \tag{4.8}$$

where β is a constant. The simulations or different depths are shown in Fig 4.9 and the same data as in the previous case is collected in Table 4.10. As it is noted for bigger β values the error is increased but there is a bigger reduction in the number of layers needed in the computations.

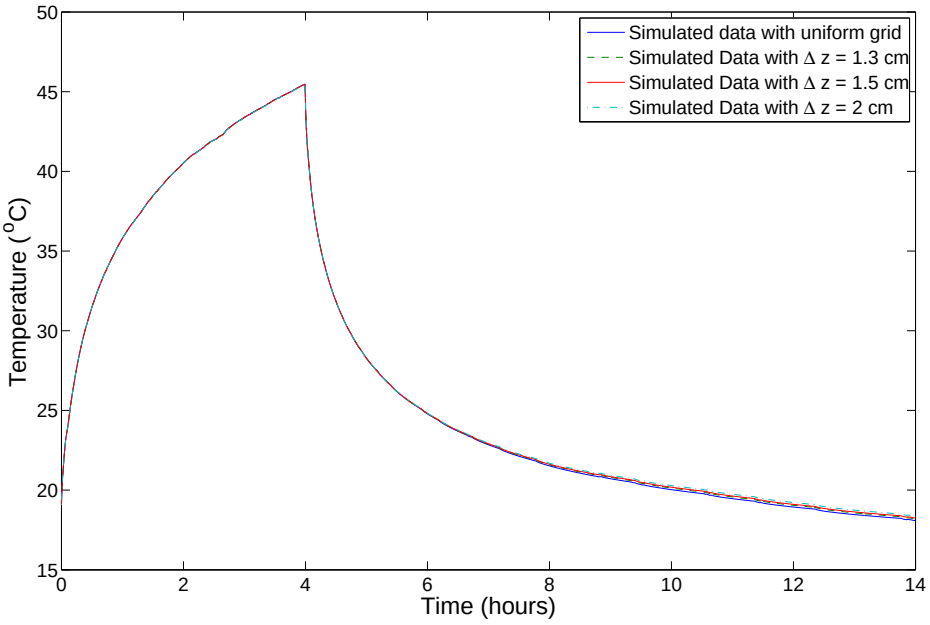


Figure 4.8: Full radiation test: comparison between the simulated data obtained using uniform grid and those obtained using non-uniform grid, increasing the Δz parameter in 20 cm and then maintaining it constant.

Finally, we have tested a third way of performing the discretization in the z direction. In this case the discretizations step Δz is maintained constant (1cm) until 20 cm and then it follows this expression:

$$\Delta z_k = \xi \ \Delta z_{k-1} \tag{4.9}$$

where ξ is a constant. In Fig 4.10 is shown the simulation using this discretization step for values of ξ. In Table 4.10 are collected the results for this non-uniform grid. It can be noted that on all non-uniform grid the differences with the uniform grid are bigger as the separation between layers grows.

The stability condition is satisfied using these non-uniform grids, as:

$$\Delta d = min\left(\Delta z_1, ...\Delta z_N, \Delta x\right) = \Delta x \tag{4.10}$$

and then the stability condition is reduced to:

$$F_0 = \frac{\alpha \Delta t}{(\Delta x)^2} < \frac{1}{6} \tag{4.11}$$

A value of $F_0 = 0.06$ is obtained, which meets the stability condition.

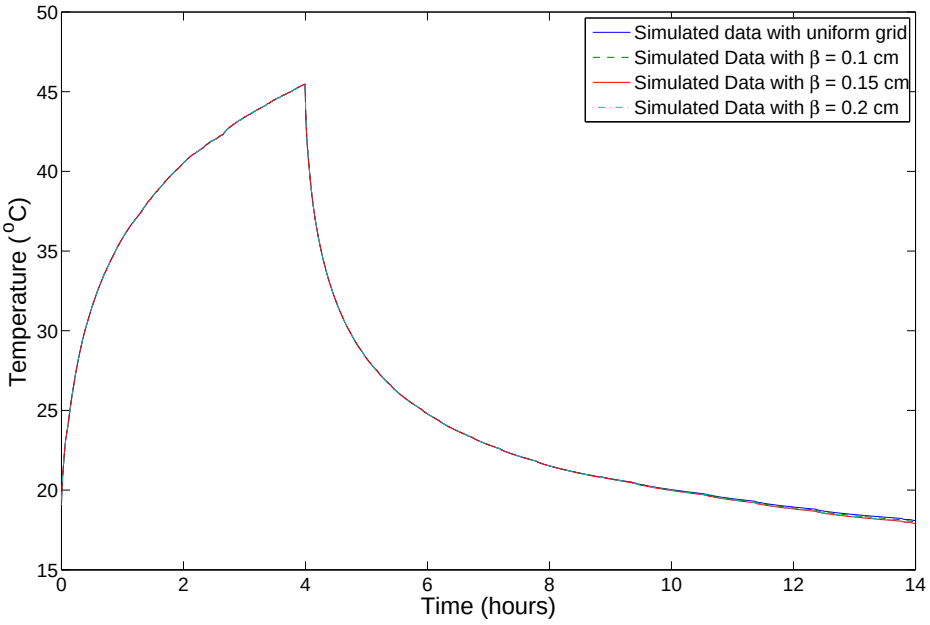


Figure 4.9: Full radiation test: comparison between the simulated data obtained using uniform grid and those obtained using non-uniform grid with different β values.

It can be noted from Figs. 4.8-4.10 that the maximum error occur near the end of the simulation, after that the solar panel was turned off. If we analyze the errors when the solar panel is heating the soil and four hours later then the errors are much smaller, see Table 4.11. As can be noted the errors are considerable smaller and as the detection algorithm works with images at the sunrise (heating process) and lasts for 1 hour, the errors introduced will be expected to be in this range.

4.4.2 Gradual heating test

In this indoor experiment the soil ($\alpha_{soil} = 6.4 \cdot 10^{-7}$ m^2/s) is heated with a solar panel but the heating process is made gradually. The intensity of the solar radiation is increased linearly from 0 to 600 $W\backslash m^2$, reaching the maximum after 2h 45min. This value is maintained for one hour and after it decreases to zero linearly, see Fig 4.11. The air temperature was estimated following this approach:

$$T_{air}(t) = T_{0,air} - T_{est} cos\left(\frac{2\pi t}{n_h}\right) \tag{4.12}$$

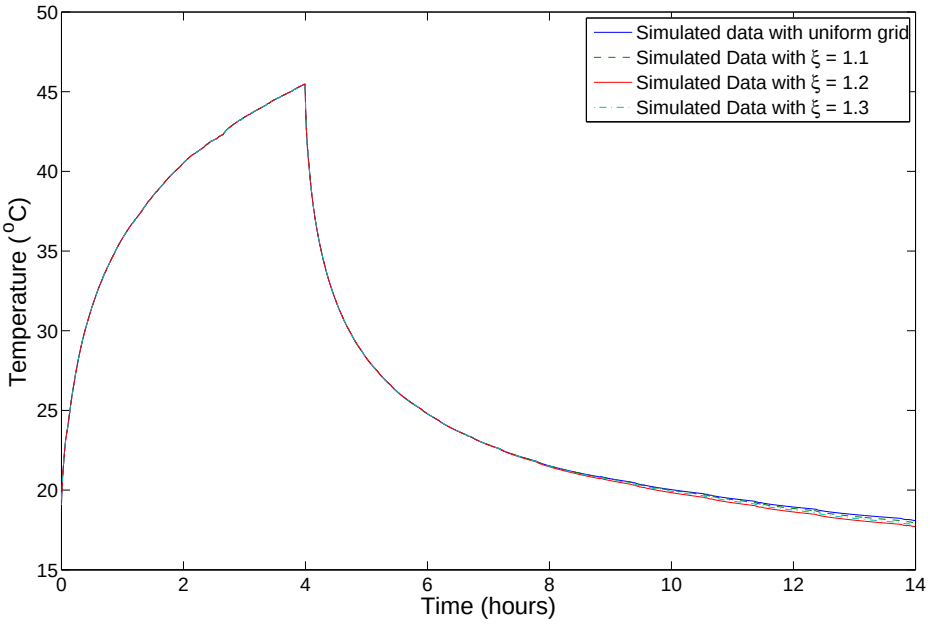


Figure 4.10: Full radiation test: comparison between the simulated data obtained using uniform grid and those obtained using non-uniform grid with different ξ values.

where t is the local time (in hours) measured from the beginning of the experiment, $n_h = 6.5$ is the number of hours that the solar panel has been radiating the soil, $T_{0,air} = 25$ ℃ and $T_{est} = 5$ ℃. In this experiment there is a shallowly buried landmine ($\alpha_{mine} = 2.64 \cdot 10^{-7}$) covered with 3 mm of sand and the perturbation that the mine introduces in the thermal behavior of the soil is observed. Similarly to the full radiation test, the values of the convective heat transfer coefficient was set to 5 W/m^{2}K. The temperatures given by probes at different depths and a later extrapolation process was used to initialize the temperatures in the volume.

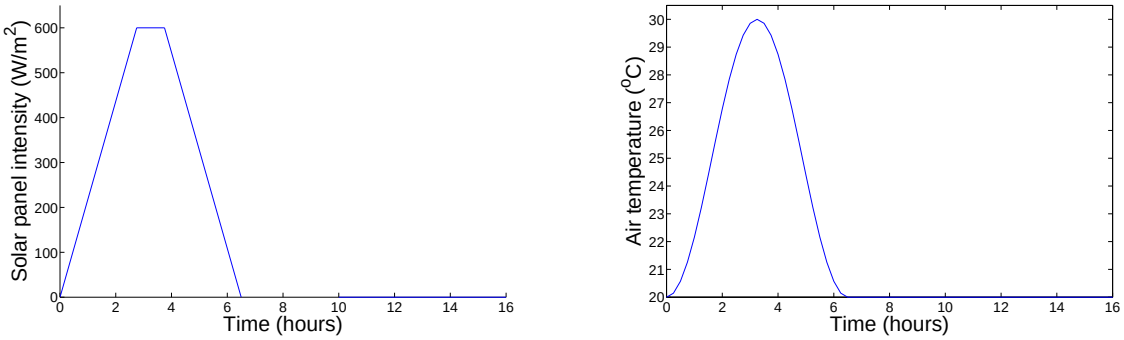
Similarly to the previous experiment, we will analyze now the effects of using integer arithmetic on the FPGA implementation and the use of non-uniform-grids.

Gradual heating test: integer arithmetic

In this case we will focus the study on the surface temperature distributions. The volume considered is composed of $100 \times 100 \times 40$ nodes, and the same spatial and discretization parameters used in the full radiation test are considered ($\Delta x = 1$ cm and $\Delta t = 10$ s). The

Table 4.10: Full radiation test: Results obtained using non-uniform grids.

		Reduction	Maximum error (℃)	TO
Δz (cm)	1.3	13 %	0.10	0.76
	1.5	18 %	0.1	0.55
	2	25 %	0.28	1.12
β (cm)	0.10	18 %	0.08	0.44
	0.15	22 %	0.14	0.64
	0.20	25 %	0.18	0.72
ξ	1.1	22 %	0.14	0.64
	1.2	30 %	0.27	0.9
	1.3	35 %	0.38	1.08



(a) Radiation with the solar panel

(b) Air temperature

Figure 4.11: Values of the boundary conditions for the gradual heating test.

mine in this experiment has a diameter of 8 cm and a height of 3 cm, thus, it is modeled as a volume of $8 \times 8 \times 3$ nodes. The comparison between the data simulated with double precision and those obtained with the FPGA using integer arithmetic is shown in Fig 4.12 both for sand and mine. The temperature values for the sand has been taken in a zone far away of the location of the mine, which its center located at x = y = 25 cm, to avoid mine perturbations, more specifically the temperature of the sand was given at position x = y = 80 cm.

As can be seen the agreement between the simulations and the experimental data is quite good for both mine and sand. The maximum differences between the data obtained

Table 4.11: Full radiation test: Results obtained using non-uniform grids when the four hours of heating and the next four hours are analyzed.

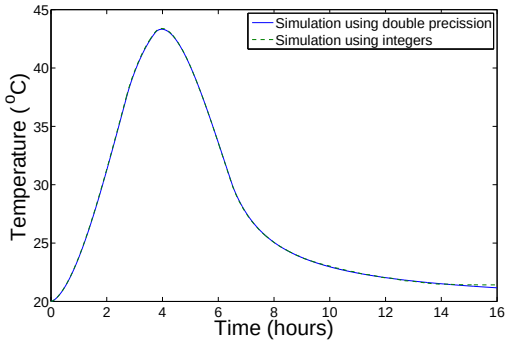
		Reduction	Maximum error (℃)	TO
Δz (cm)	1.3	13 %	0.07	0.54
	1.5	18 %	0.1	0.55
	2	25 %	0.18	0.72
β (cm)	0.10	18 %	0.004	0.02
	0.15	22 %	0.006	0.03
	0.20	25 %	0.01	0.04
ξ	1.1	22 %	0.006	0.03
	1.2	30 %	0.03	0.1
	1.3	35 %	0.07	0.2

using double precision and those obtained using integer arithmetic are less than 0.1 ℃, which are acceptable for the landmine detection application. Fig. 4.12(c) shows the thermal contrast soil-mine. It can be seen how the temperature above the mine is higher than the unperturbed soil when the solar panel is heating the soil. This is caused by the smaller thermal diffusivity coefficient of the mine compared to that of the sand, thus the mine blocks the heat transfer producing a higher temperature at the surface. As the solar panel intensity decreases the system evolves to an equilibrium point and then the thermal contrast tends to zero. Once the solar panel has been turned off the bottom of the mine blocks the heat transfer of the soil to the surface, which causes that the temperature above the mine lower than the unperturbed soil. Finally, the equilibrium is reached and the thermal contrast is, again, zero.

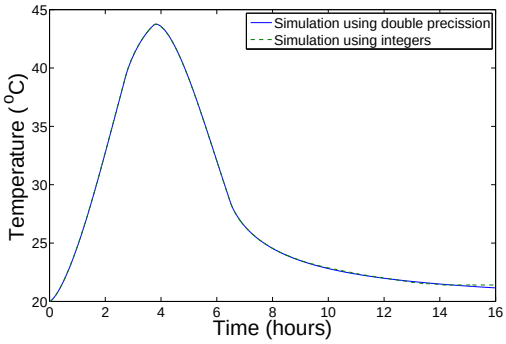
The same speedup is achieved in this test, 34, as the FPGA stops the processing every 20 minutes of simulation time to allow reading data from the HOST.

Gradual heating test: Non-uniform grid

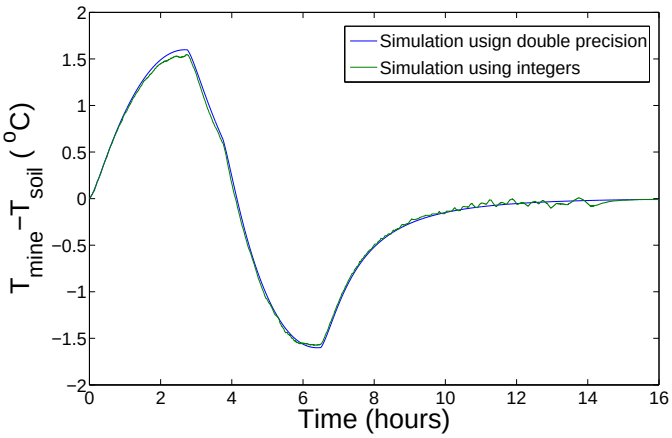
We will perform now an analysis of the use of different grid variations on the z-direction and its influence on the simulations. All the simulations have been carried out using MATLAB (double precision). Similarly to the full-radiation test we will analyze 3 different



(a) Unperturbed sand



(b) Sand with mine underneath



(c) Thermal contrast mine-soil

Figure 4.12: Gradual heating test: comparison between the simulated data obtained using double precision and integer arithmetic and thermal contrast soil-mine

non-uniform grids schemes. The uniform grid has 40 layers on the z-direction and all non-uniform grids remains constant until 20 cm, varying the grid from 20 cm.

In Fig. 4.13 we can see a comparison between the data obtained with the uniform grid and this non-uniform grid scheme where the Δz value changes at 20 cm and then remains constant. In Table 4.12 the achieved reduction with each value of Δz, the maximum difference between the temperatures obtained with the uniform grid and those obtained with this non-uniform grid scheme are summarized. Fig. 4.14 shows the results obtained in the simulation using the non-uniform grid given by Eq. (4.8). It can be noted that although the differences between the temperatures with the uniform grid and those obtained with the non-uniform grids, mainly for the high value of β, the differences between the temperature

Table 4.12: Gradual heating test: Results obtained using non-uniform grids.

		Reduction	Maximum error (℃)	TO
Δz (cm)	1.3	13 %	0.12	0.92
	1.5	18 %	0.17	1.05
	2	25 %	0.22	0.88
β (cm)	0.10	18 %	0.18	1.00
	0.15	22 %	0.30	1.36
	0.20	25 %	0.37	1.45
ξ	1.1	22 %	0.24	1.09
	1.2	30 %	0.36	1.2
	1.3	35 %	0.50	1.42

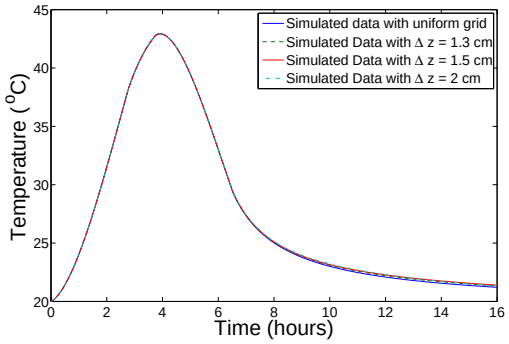
of the soil and the mine remain unperturbed, see Fig. 4.14(c). Table 4.12 shows the maximum differences and the reduction of the number of layers achieved for each value of β.

Finally, Fig. 4.15 shows the results obtained in the simulation using the non-uniform grid given by Eq. (4.9). It can be noted that although the differences between the temperatures with the uniform grid and those obtained with the non-uniform grids, mainly for the high value of β, the differences between the temperature of the soil and the mine remain unperturbed, see Fig. 4.15(c).

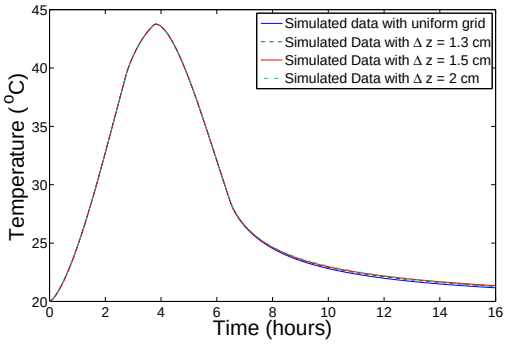
From Tables 4.10, 4.12 and from the errors due to the use of integer arithmetics in the FPGA it can be seen that both scheme are useful to reduce the computing without altering the precision of the computations.

4.4.3 Natural heating test

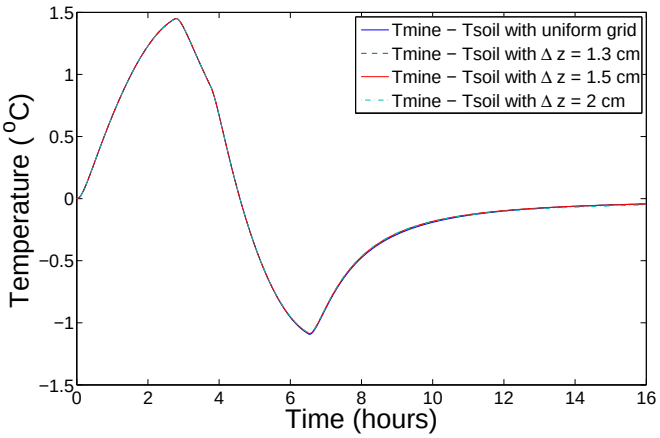
This tests corresponds with the outdoor scenario described at the beginning of this chapter. In this section we will consider an experiment in which the soil is heating by the sun radiation (natural heating). The thermal properties for the mine and the soil for these examples can be seen in Table 4.13. The test images were acquired at the TNO facilities in April 6, 2001. We will analyze the temporal interleave between the 9:41 am and 10:41 am; it is at the sunrise when the thermal contrast between mine and soil temperatures is



(a) Unperturbed sand.



(b) Perturbed sand.

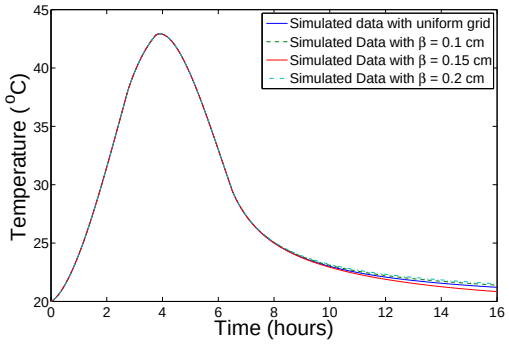


(c) Thermal contrast mine-soil.

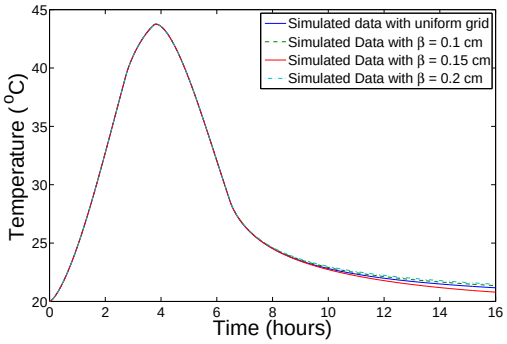
Figure 4.13: Gradual heating test: Comparison between the simulated data obtained using uniform grid and those obtained using non-uniform grid for both unperturbed soil and mine. In (c) it is represented the temperature difference between soil and mine.

higher and also is easier to initialize the temperatures, (López et al., 2004; Thanh et al., 2006), due to the fact that during the night the temperature of the soil tends to become uniform. In this experiment the values of air temperature and solar radiation intensity were measured.

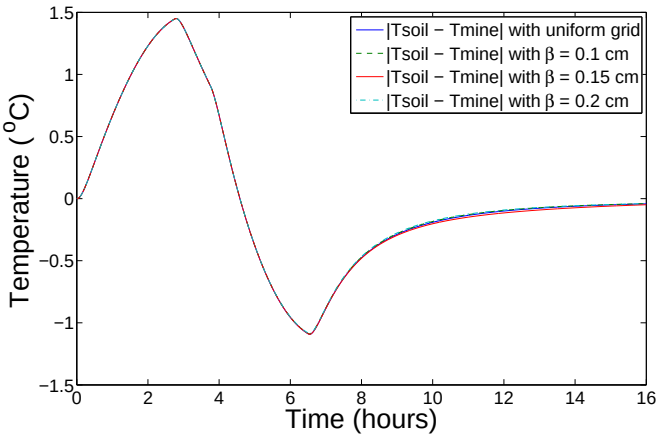
In this experiment there are no internal probes to measure the temperature in depth. Thus, to initialize the temperature of the volume of soil we have to use the temperatures at the surface, given by the IR images, to extrapolate the temperature of points below the surface. Taking into account that during the night the absence of the sun makes the temperature of the soil more uniform we have assumed that there is a variation of 4℃



(a) Unperturbed sand.



(b) Unperturbed sand.

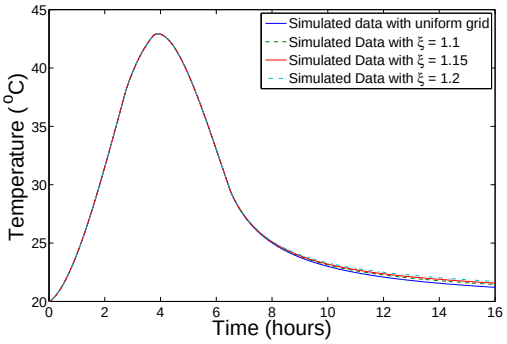


(c) Thermal contrast mine-soil.

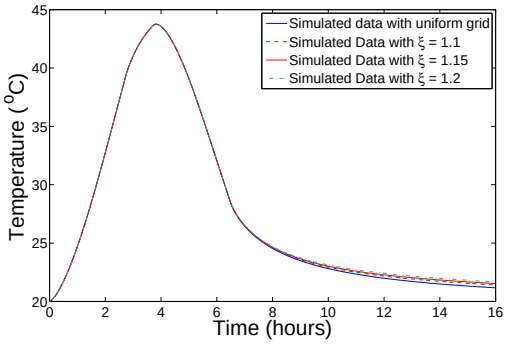
Figure 4.14: Gradual heating test: Comparison between the simulated data obtained using uniform grid and those obtained using non-uniform grid for both unperturbed soil and mine. In (c) it is represented the temperature difference between soil and mine.

between the surface and the depth of 50cm. Considering this fact a linear interpolation is made to obtain the initial temperatures of internal nodes, (López, 2003).

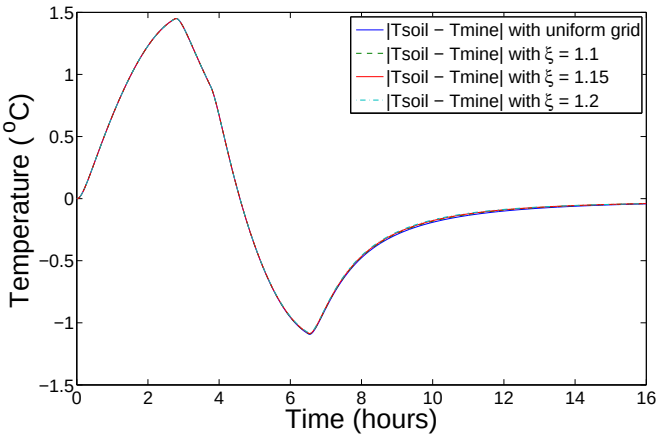
To illustrate the behavior of the thermal model we study in depth two examples. The parts of the sand lane used in these two examples can be seen in Fig. 4.16. In the Example 1 there are three surface laid mines and one shallowly buried mine, while in the Example 2 there is a surface laid mine, one shallowly buried mine and two mines buried at 6 cm.



(a) Unperturbed sand.



(b) Unperturbed sand.



(c) Thermal contrast mine-soil.

Figure 4.15: Gradual heating test: Comparison between the simulated data obtained using uniform grid and those obtained using non-uniform grid for both unperturbed soil and mine. In (c) it is represented the temperature difference between soil and mine.

Example 1

Now we will analyze the simulation of the example 1, the initial temperature of the considered example is shown is Fig. 4.17, where the four mines presented the image are indicated. To validate the FPGA implementation we will follow the full detection process proposed in (López, 2003). First we will simulate the thermal model assuming mine absence, i.e. homogeneous soil. Fig. 4.18 shows the temperature at 10:41 obtained with MATLAB and with the FPGA in this situation, it can be seen that the temperature is almost uniform in the piece of soil, according to the mine absence assumption. The differences between the

Table 4.13: Thermal properties of the mine and soil for the natural heating test

	Dry sand	Antipersonnel mine
$\rho(kg/m^3)$	1650	1350
$c_p(J/kgK)$	710	1120
$k(W/mK)$	0.75	0.4

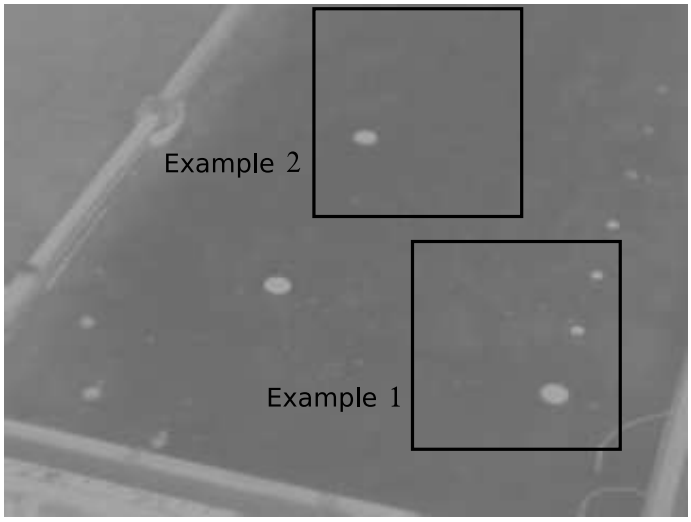
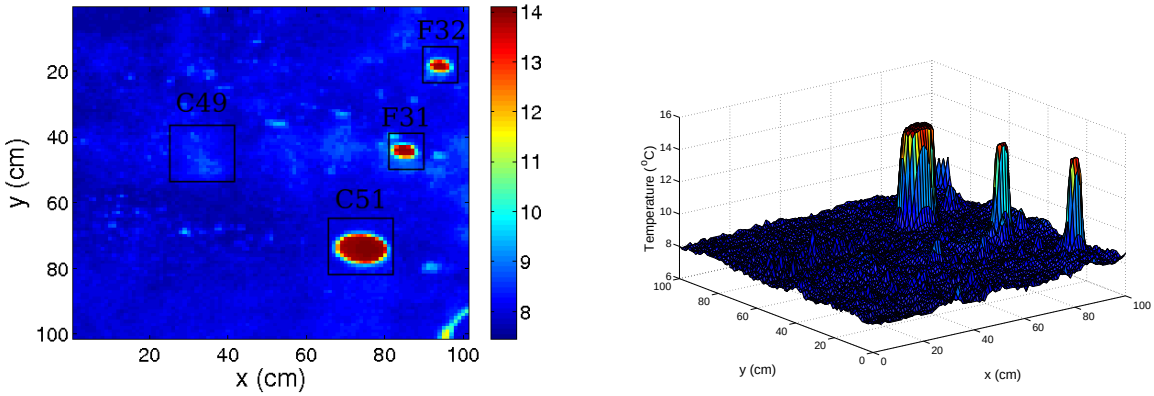


Figure 4.16: Natural heating: IR image where the pieces of soil used to perform the simulations in both examples are indicated.



(a) Initial temperature at 09:41.

(b) 3D representation of the initial temperature

Figure 4.17: Natural heating ex1: Initial temperature at 09:41.

temperatures obtained with the FPGA and MATLAB is lower than 0.15 ℃.

Now the a process to detect the possible targets presented in the soil is run, which provides a classification map of the image. The results of the classification phase can be seen in Fig 4.19 for both MATLAB and FPGA simulations. The obtained mask are the same, which demonstrates the validity of the FPGA implementation and that the errors caused by the use of integer arithmetic (¡ 0.1 ℃) does not affect to the performance of the detection algorithm. Now we present the results of the simulations using integer arithmetic and non-uniform grids once the detection algorithm has established the properties of the objects (mines) and their buried depth. The simulation using integer arithmetic can be seen in Fig. 4.20. It can be note the low error introduced (less than 0.1 ℃. In Figs. 4.22-4.24 we can see the simulations performed using the three non-uniform grids.

Example 2

In this example we have one surface laid mine, corresponding to mine C37, one shallowly buried mine, corresponding to mine C36 and two buried mines at 6cm, corresponding to B59 and B60 in Fig. 4.3. Fig. 4.25 shows the temperatures of the soil, given by the IR images, at 09:41 and its 3D representation, where it can be clearly seen that there are two mines that cause a perturbation in the temperature of the soil. The considered image have a size of 100×100 pixels.

Similarly to the previous examples first we will analyze the use of integer arithmetic with the FPGA using uniform grid. Thus, the resulting grid has a size of 100×100×50, considering $\Delta x = \Delta y = \Delta z = 1cm$. Fig. 4.26 shows a comparison between the data obtained at 10:41 am using double arithmetic and integer arithmetic. The differences between the temperatures obtained using double precision and integer arithmetic can be seen in Fig. 3.4.5. It can be seen how the maximum difference reaches 0.15 ℃ while the thermal contrast soil-mine is around 3 ℃. Thus, the error introduced due to the integer arithmetics in the FPGA does not affect to the precision of the system.

Now we will show the simulation obtained using the 3 non-uniform grids for this example. In this case the grid in the z-direction remains constant until 20cm and then it varies according to the schemes previously introduced. It also can be noted that in all situations the differences between using uniform and non-uniform grid are in the order of 10^{-5} ℃. In this case the use of non-uniform grids produces lower differences with the uniform grid simulation as in the previous examples because the simulation time is lower

Table 4.14: Natural heating: Reduction in the number of nodes using non-uniform grids.

	Δ z (cm)			β (cm)			ξ		
	1.3	1.5	2.0	0.10	0.15	0.20	1.1	1.15	1.2
Reduction	14 %	20 %	30%	26%	32%	34%	32%	40%	46%

in this situation and the error caused by the non-uniform grid does not propagates.

4.5 Classification example

To illustrate the performance of the FPGA implementation we will consider an example from (López, 2003) where the position and nature of a buried object is done. The object $V82$ (metallic disk) was considered in this example. As was said the classification algorithm is divided in two parts. In the first step the algorithm obtains the position of the buried object and classifies it as mine or non-mine; this is a quasi-inverse procedure because the position at which the object can be located are restricted to grid positions. In the second step a full-inverse procedure is run to obtain the nature of the objects classified as non-mine, through the obtention of their thermal properties (value of α). The first step takes 80 iterations and after this the object was classified as non-mine and its location was obtained. After this the full-inverse process is carried out. The evolution of the process can be seen in Fig. 4.31. As can be noted the stopping criteria finish the procedure after 230 iterations and a value of *alpha* similar to that of the metal is obtained ($170 \cdot 10^{-7}$ m^2/s).

The whole classification procedure, in this example, involves the computing of 310 different configurations, which takes 15 hours in a PC. On the other hand, this full process takes 28 minutes on the FPGA, which demonstrates the utility of the FPGA hardware solver.

Table 4.15: Comparison of the speedups using different FPGAs. The speedups for the Virtex-4 and Virtex-5 and that of the Virtex-II using 16 memory banks are estimations.

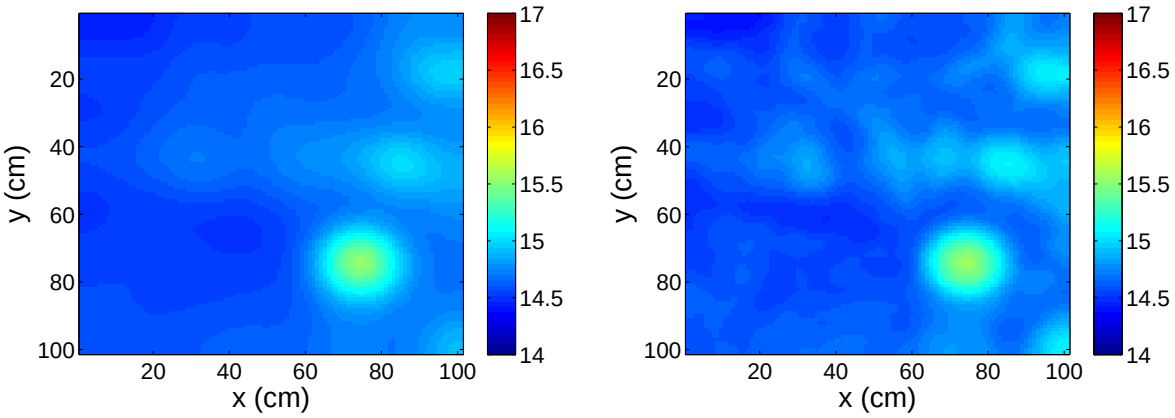
FPGA	Speedup (8 memory banks)	Speed-up (16 Memory banks)
$Virtex - II$	34	60
$Virtex - 4$	86	160
$Virtex - 5$	110	210

4.6 Performance of the FGPA-based thermal model hardware solver

The total speedup factor of the FPGA mixed design with respect to a Pentium IV 3 GHZ single precision implementation (C++) is 34, which demonstrates the validity of the approach as an efficient heat equation solver and hardware accelerator. The bottleneck of the system is the access to the memory banks. An estimation of the speedup that could be achieved introducing more memory banks can be done by taking into account two issues, on the one hand the number of free slices to introduce new PE and, on the other hand, the number of available $IOBs$ to connect new memory banks to the FPGA. Concerning to the first issue, each PE consumes 600 slices and assuming a 10% of over cost to do the routing we have estimated that at least 40 new PE could be introduced in the system due to the limitations in the number of slices. On the other hand, the introduction of a new memory bank consumes 59 $IOBs$ in the FPGA. As there are still 487 $IOBs$ available, eight new memory banks could be connected to the FPGA. As can be noted, the number of possible PE that could be added to the system is much bigger than the number of memory banks that could be introduced. Therefore, with the current hardware, a maximum of eight memory banks and eight PE could be added. With this hypothetical scenario we have estimated a speedup factor of 68. We have also estimated the theoretical speedup that could be achieved using a Virtex-4 and a Virtex-5 instead of the Virtex-II RC2000 card's FPGA. We have synthesized the processing core changing the FPGA and we have obtained a clock of 140 MHz for a Virtex-5 and of 95 MHz for a Virtex-4, whereas for the Virtex-II a clock of 50 MHz is achieved. The resulting speedups in each case are collected in Table 4.15. As can be seen, in the case of a Virtex-5 with 16 memory banks, the total speedup factor would be 210, which clearly demonstrates the feasibility of a FPGA implementation as a hardware accelerator of the heat equation solver.

4.7 Summary

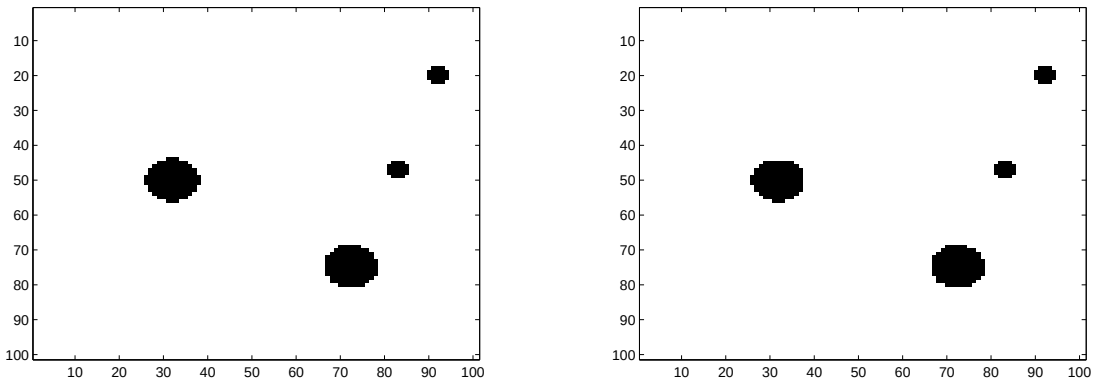
In this chapter we have probed the validity of the thermal model FPGA implementation, where the use of integer arithmetics provides enough precision to perform the mine detection. We have also shown that due to the characteristics of the thermal transfer in the soil the number of layers involved in the computations can be reducing by increasing the grid size in the z direction for higher depths, where temperatures variations are very low. The excessive processing time required by the detection algorithm is reduced, allowing to develop a system for being used in on-field applications.



(a) Temperature at 10:41 (MATLAB)

(b) Temperature at 10:41 (FPGA)

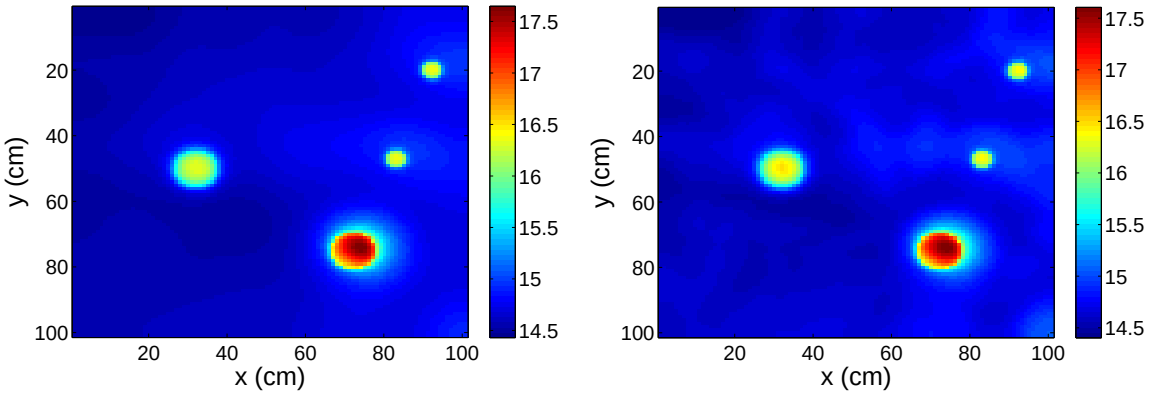
Figure 4.18: Natural heating ex1: Temperature at 10:41 obtained with MATLAB (double precision) and FPGA (integer arithmetic) assuming mine absence.



(a) Mask of detected targets (MATLAB)

(b) Mask of detected targets (FPGA)

Figure 4.19: Natural heating ex1: Mask of detected targets for MATLAB and FPGA simulations.



(a) Temperature at 10.41 (MATLAB)

(b) Temperature at 10.41 (FPGA)

Figure 4.20: Natural heating ex1: Temperatures of the soil at 10:41 obtained through simulation.

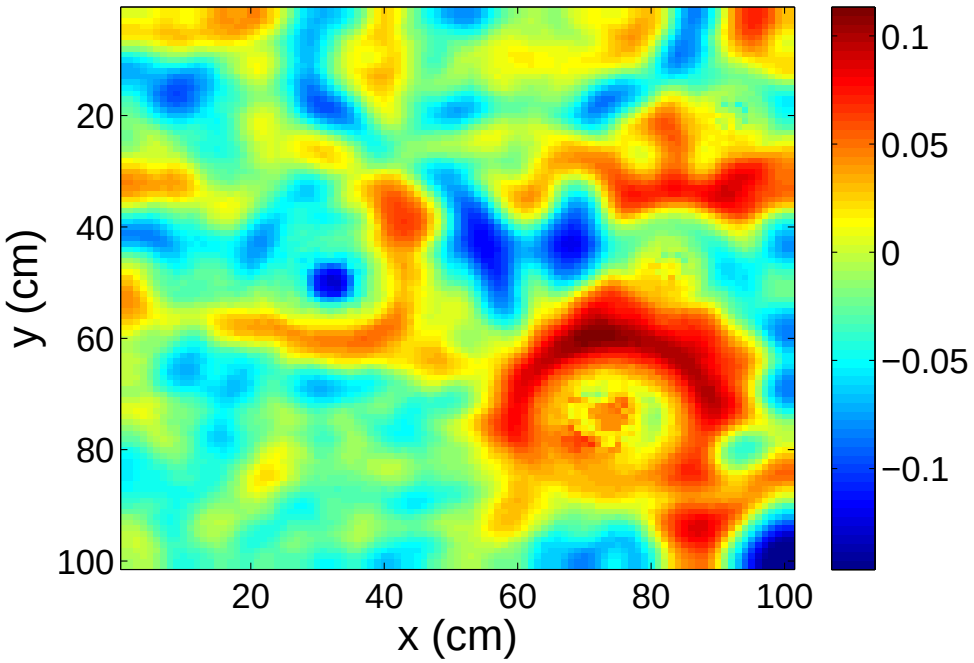
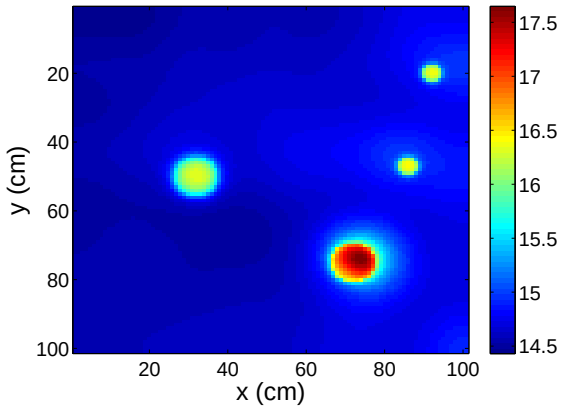
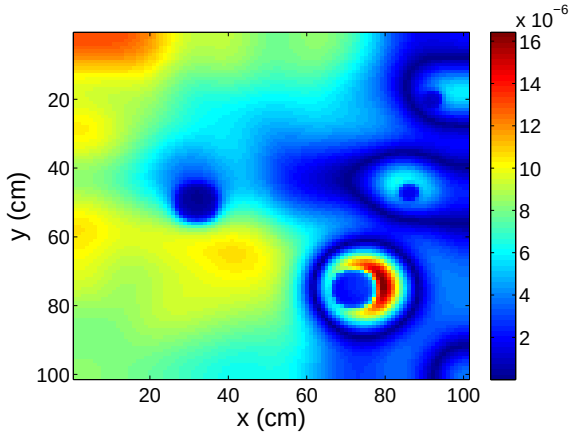


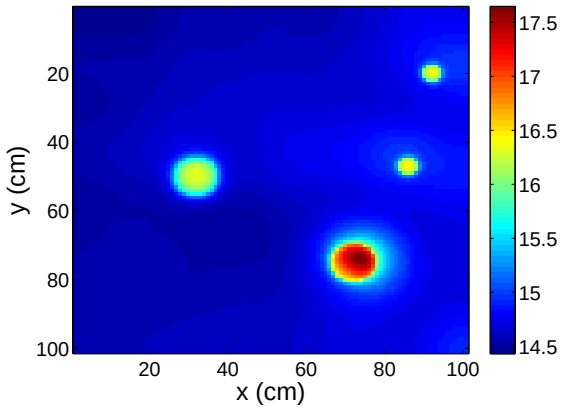
Figure 4.21: Natural heating ex1: Differences between the temperatures obtained using double precision and integer arithmetic



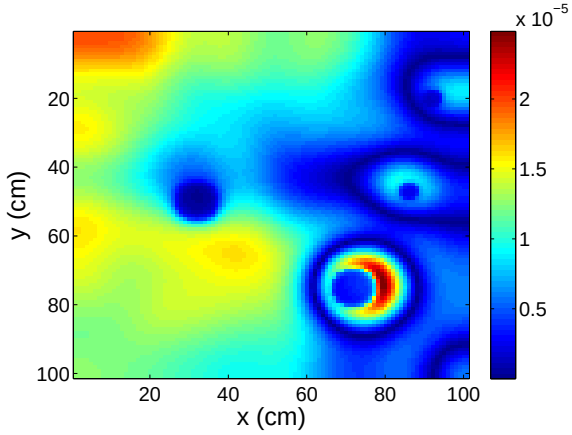
(a) Temperature at 10:41 with $\Delta z = 1.3$ cm



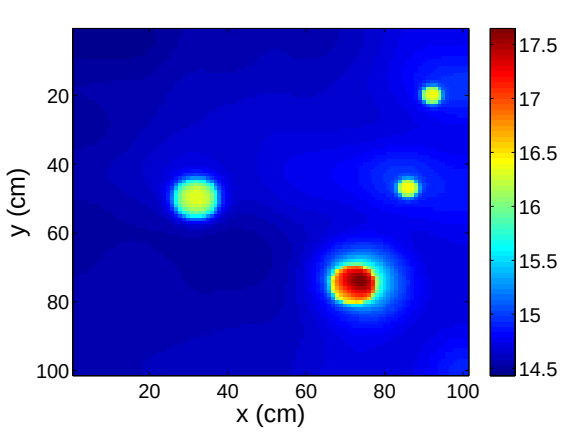
(b) Temperature difference with $\Delta z = 1.3$ cm



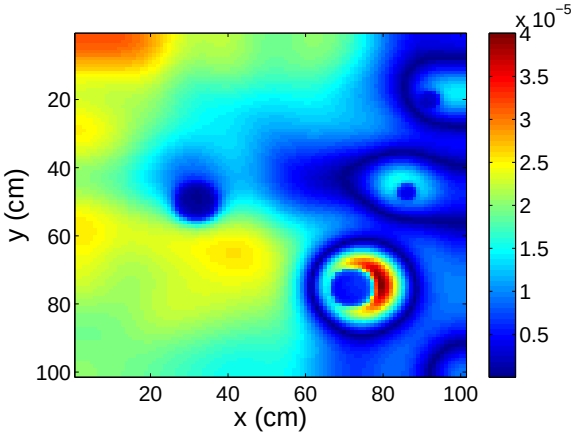
(c) Temperature at 10:41 with $\Delta z = 1.5$ cm



(d) Temperature difference with $\Delta z = 1.5$ cm

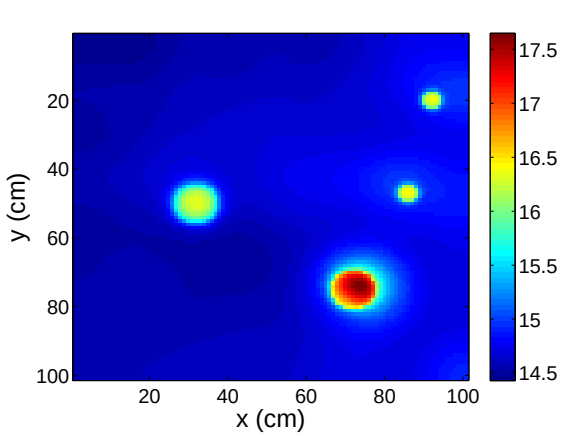


(e) Temperature at 10:41 with $\Delta z = 2$ cm

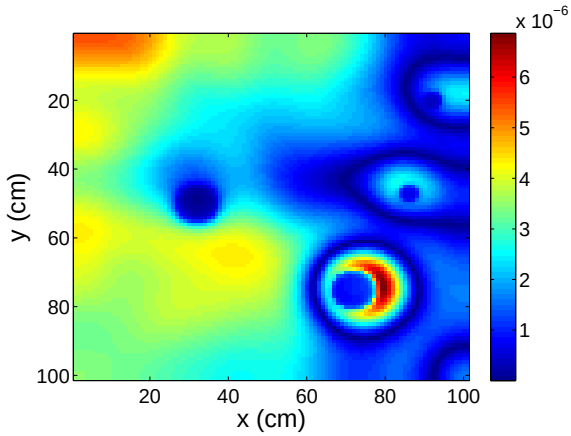


(f) Temperature difference with $\Delta z = 2$ cm

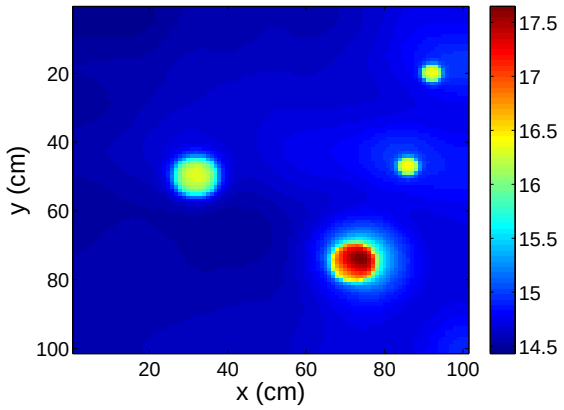
Figure 4.22: Natural heating ex1: Simulations obtained at 10:41 using non-uniform grid scheme 1 and temperature differences between the simulations using uniform and non-uniform grid.



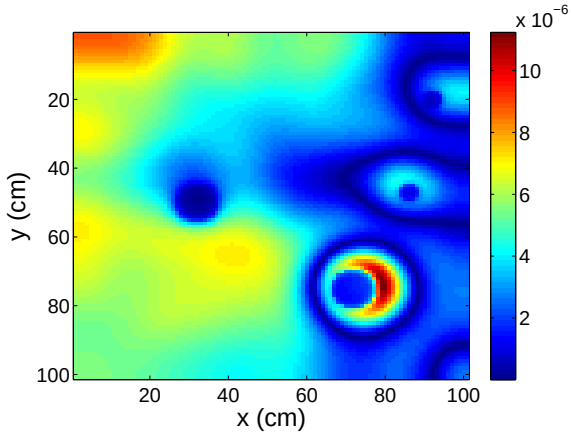
(a) Temperature at 10:41 with $\beta = 0.10$ cm



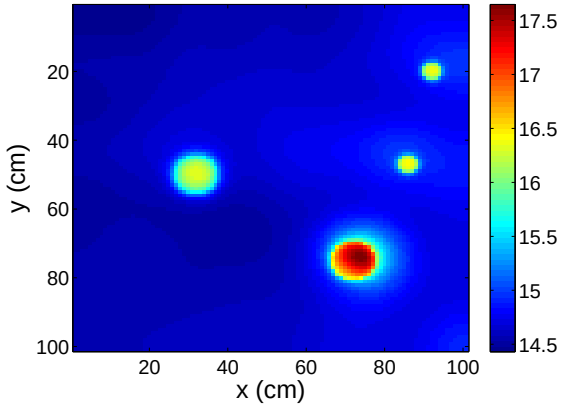
(b) Temperature difference with $\beta = 0.10$ cm



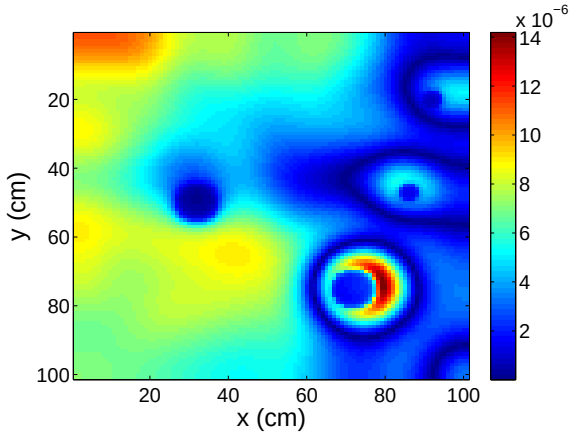
(c) Temperature at 10:41 with $\beta = 0.15$ cm



(d) Temperature difference with $\beta = 0.15$ cm

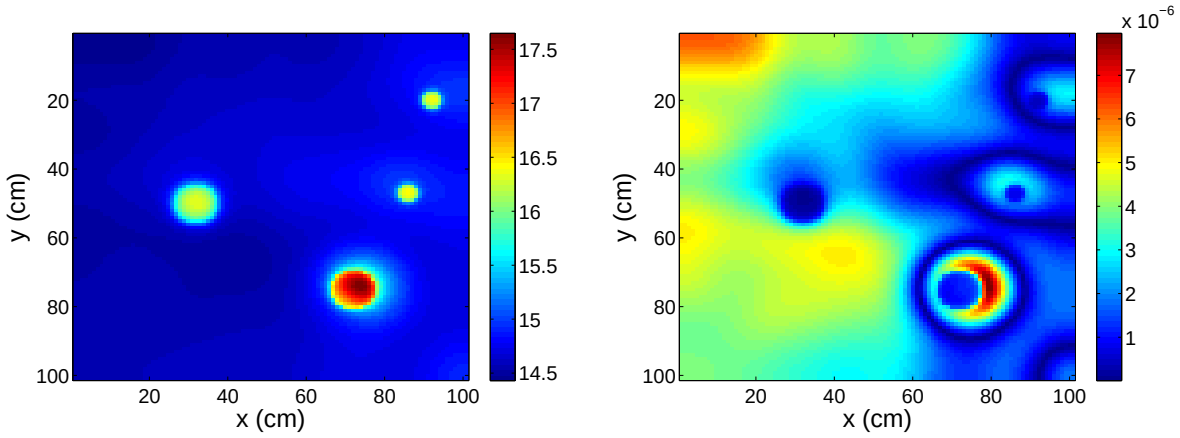


(e) Temperature at 10:41 with $\beta = 0.2$ cm



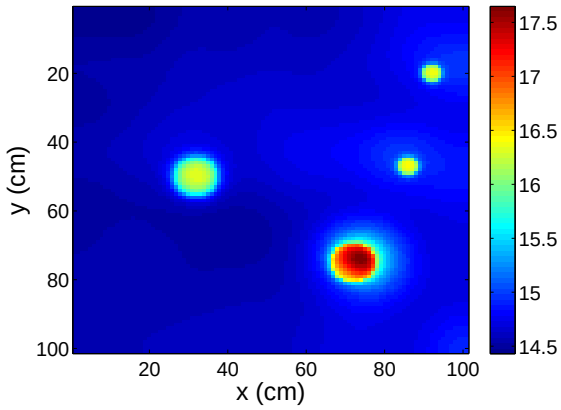
(f) Temperature difference with $\beta = 0.2$ cm

Figure 4.23: Natural heating ex1: Simulations obtained at 10:41 using non-uniform grid scheme 2 and temperature differences between the simulations using uniform and non-uniform grid.

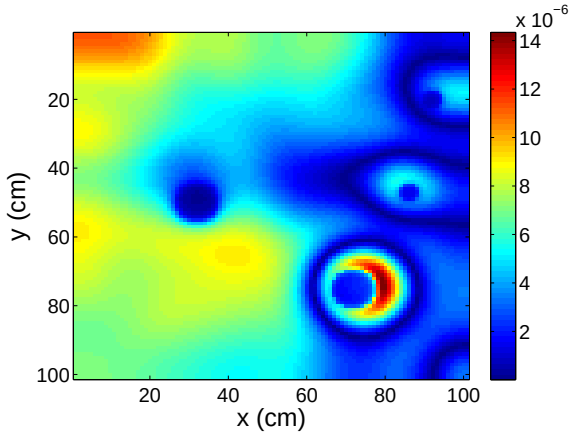


(a) Temperature at 10:41 with $\xi = 1.10$

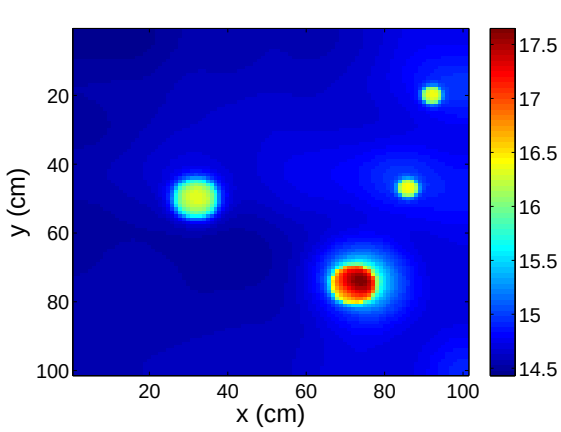
(b) Temperature difference with $\xi = 1.10$



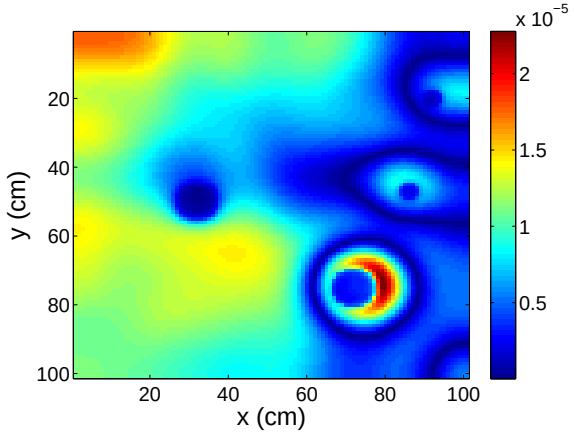
(c) Temperature at 10:41 with $\xi = 1.20$



(d) Temperature difference with $\xi = 1.20$

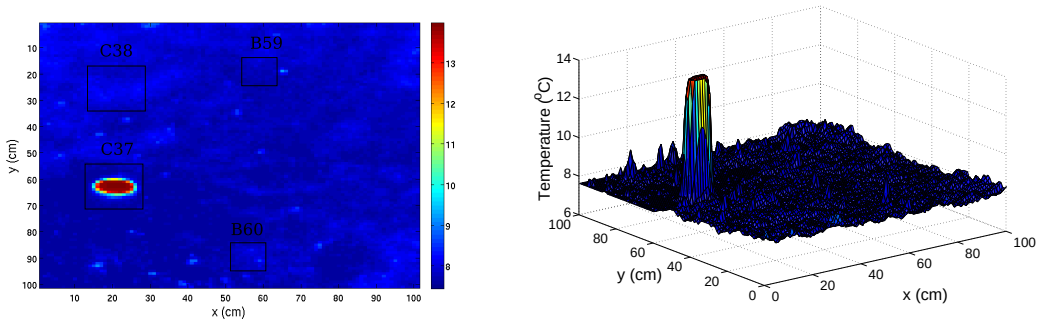


(e) Temperature at 10:41 with $\xi = 1.3$



(f) Temperature difference with $\xi = 1.3$

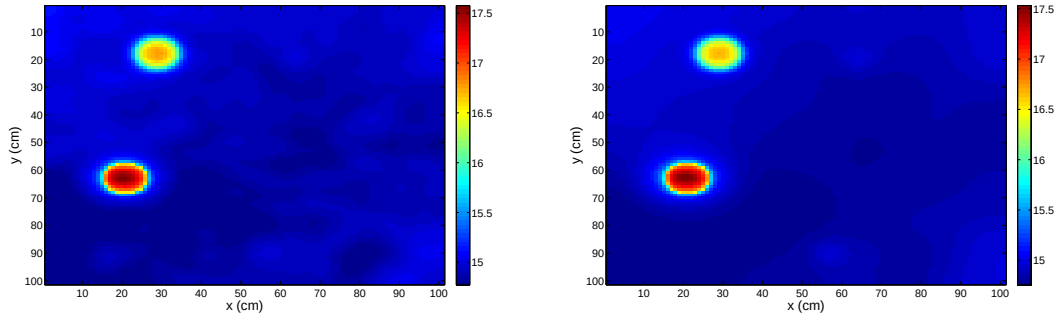
Figure 4.24: Natural heating ex1: Simulations obtained at 10:41 using non-uniform grid scheme 3 and temperature differences between the simulations using uniform and non-uniform grid.



(a) Temperature at 09.41.

(b) 3D plot of temperature at 09:41.

Figure 4.25: Natural heating ex2: Temperatures of the soil at 09:41 given by the IR camera.



(a) Temperature at 10.41 (MATLAB)

(b) 3D Temperature at 10.41 (FPGA)

Figure 4.26: Natural heating ex2: Temperatures of the soil at 10:41 obtained through simulation.

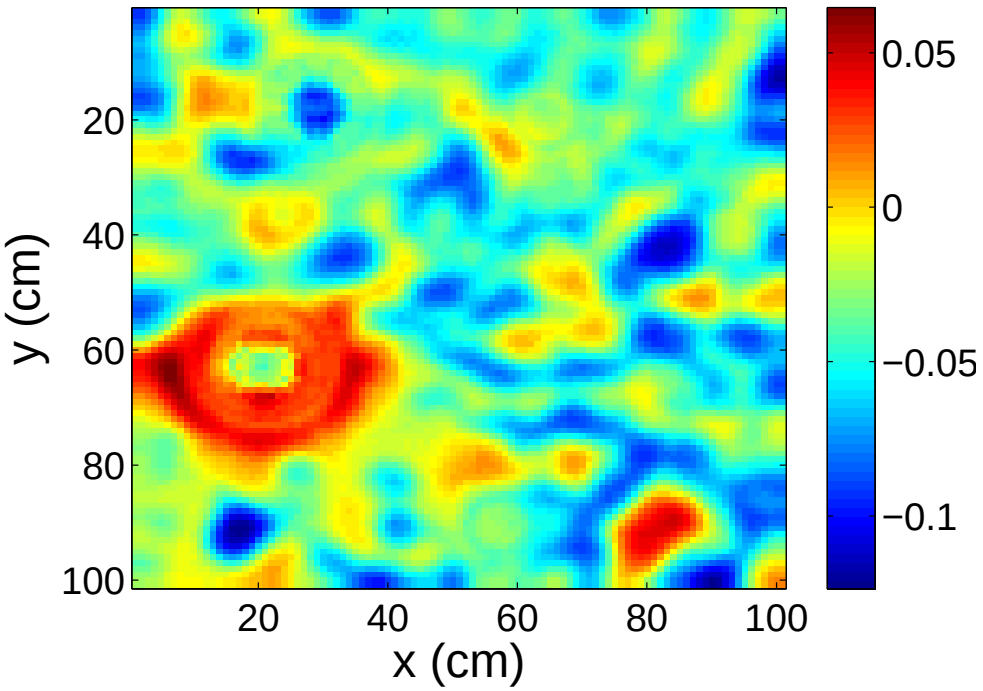
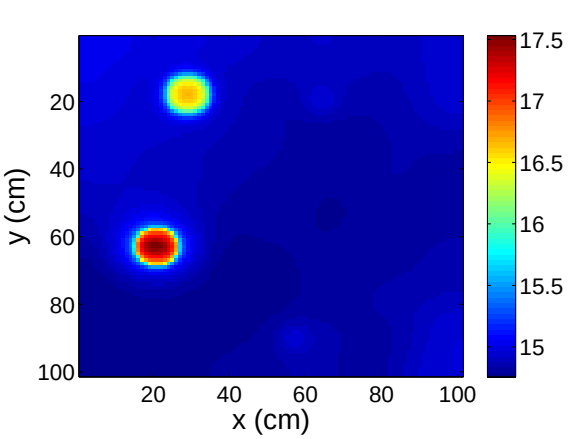
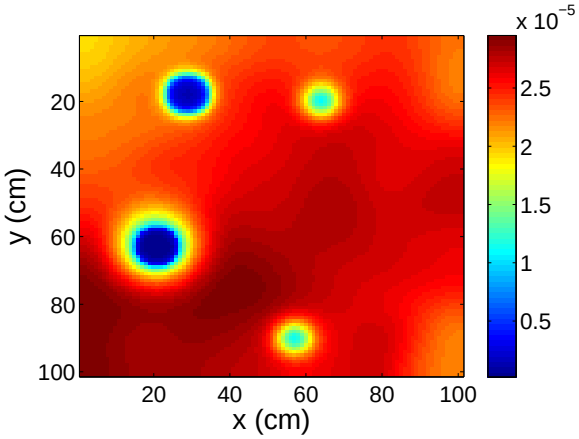


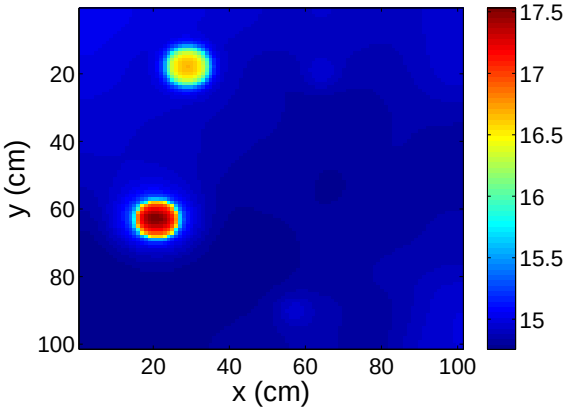
Figure 4.27: Natural heating ex2: Differences between the temperatures obtained using double precision and integer arithmetic



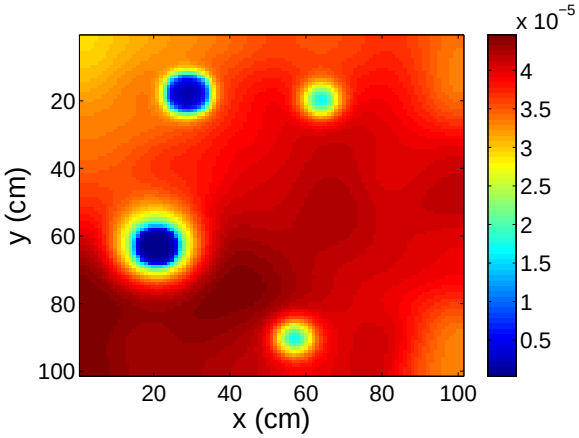
(a) Temperature at 10:41 with $\Delta z = 1.3$ cm



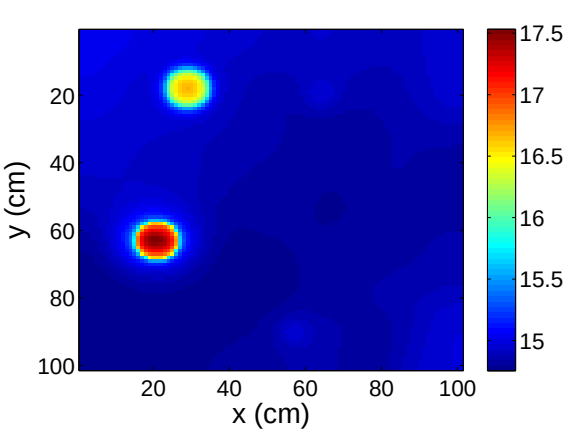
(b) Temperature difference with $\Delta z = 1.3$ cm



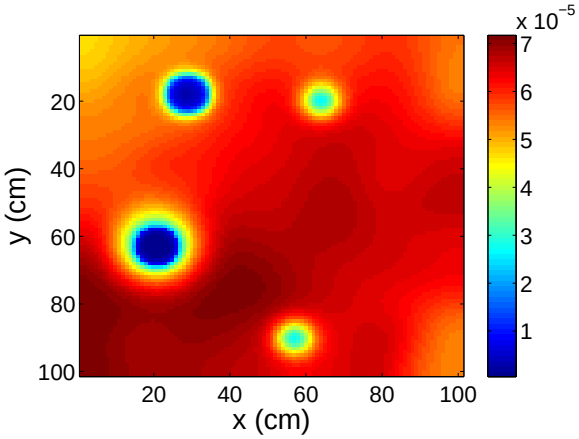
(c) Temperature at 10:41 with $\Delta z = 1.5$ cm



(d) Temperature difference with $\Delta z = 1.5$ cm

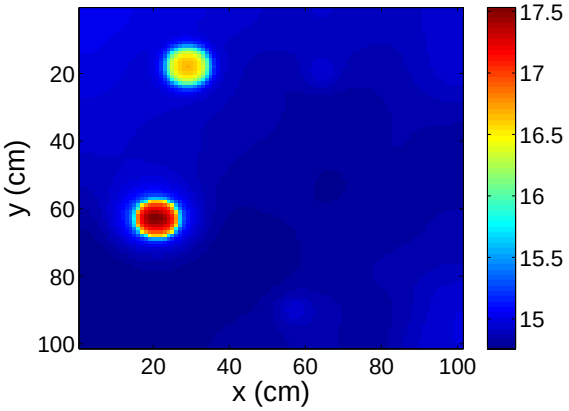


(e) Temperature at 10:41 with $\Delta z = 2$ cm

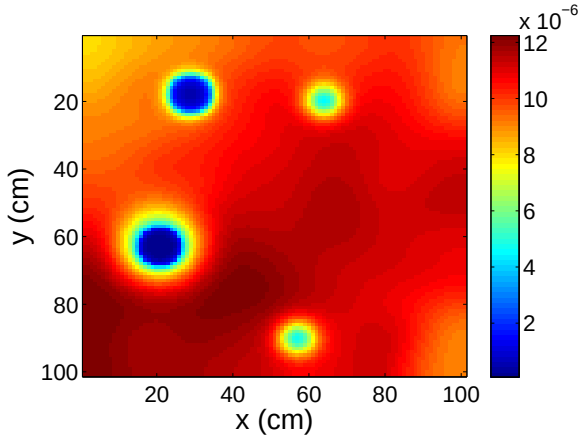


(f) Temperature difference with $\Delta z = 2$ cm

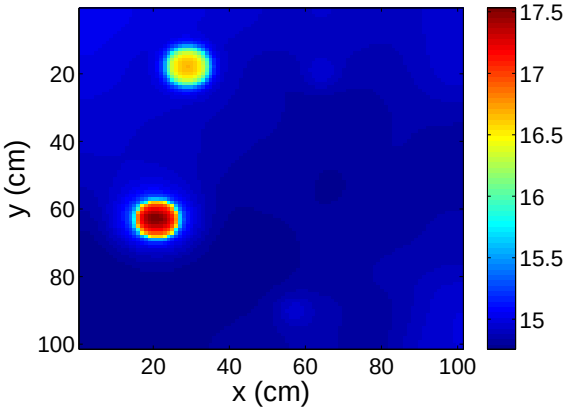
Figure 4.28: Natural heating ex2: Simulations obtained at 10:41 using non-uniform grid scheme 1 and temperature differences between the simulations using uniform and non-uniform grid.



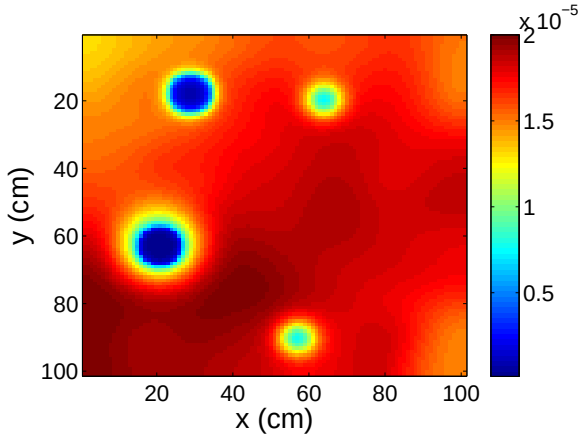
(a) Temperature at 10:41 with $\beta z = 0.10$ cm



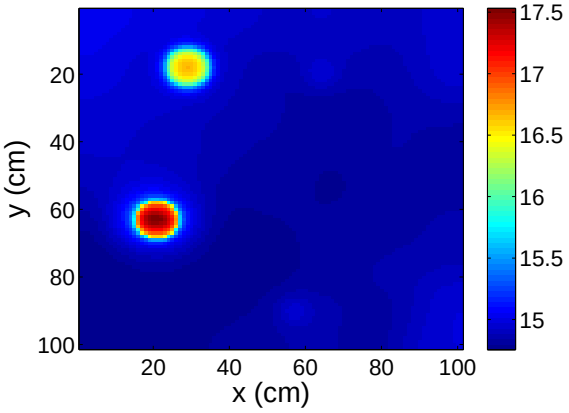
(b) Temperature difference with $\beta = 0.10$ cm



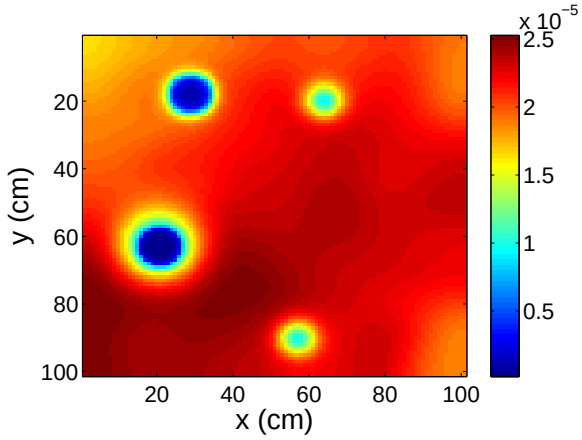
(c) Temperature at 10:41 with $\beta = 0.15$ cm



(d) Temperature difference with $\beta z = 0.15$ cm

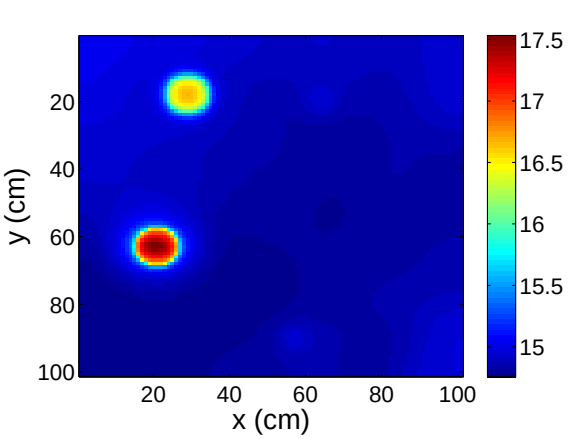


(e) Temperature at 10:41 with $\beta = 0.2$ cm

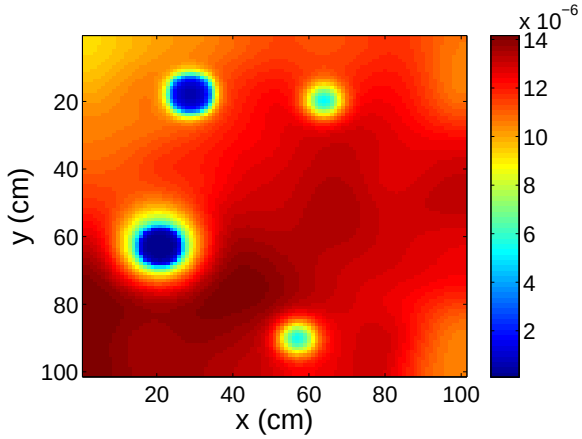


(f) Temperature difference with $\beta = 0.2$ cm

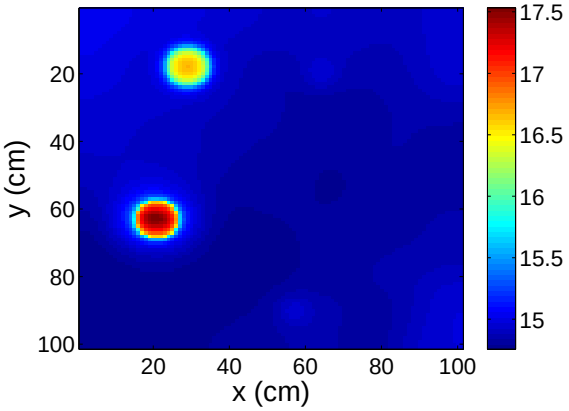
Figure 4.29: Natural heating ex2: Simulations obtained at 10:41 using non-uniform grid scheme 2 and temperature differences between the simulations using uniform and non-uniform grid.



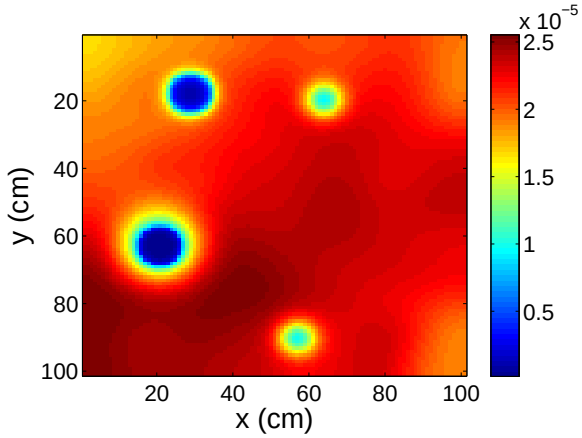
(a) Temperature at 10:41 with $\xi z = 1.10$



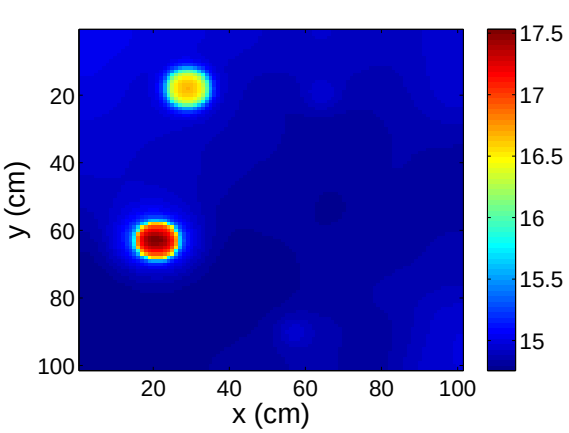
(b) Temperature difference with $\xi = 1.10$



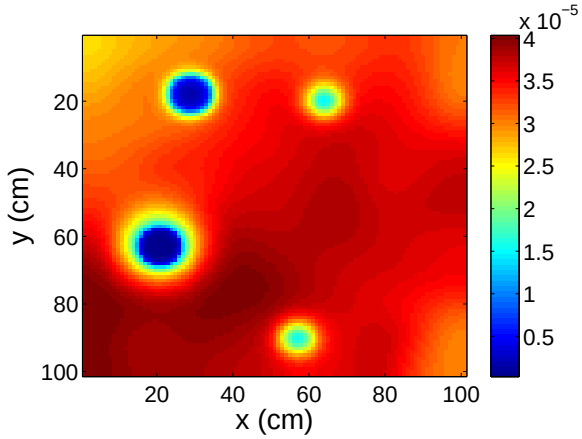
(c) Temperature at 10:41 with $\xi = 1.20$



(d) Temperature difference with $\beta z = 1.20$

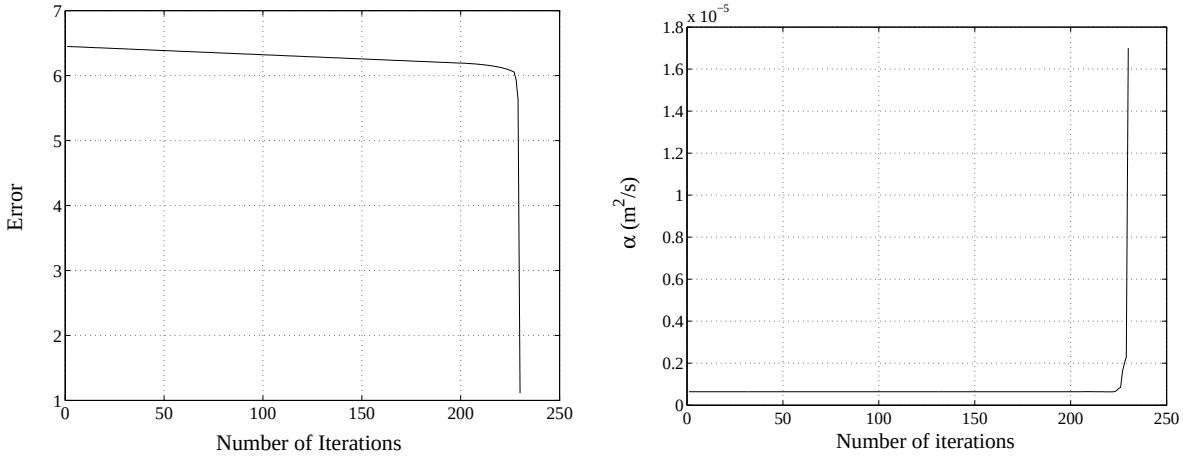


(e) Temperature at 10:41 with $\xi = 1.3$



(f) Temperature difference with $\xi = 1.3$

Figure 4.30: Natural heating ex2: Simulations obtained at 10:41 using non-uniform grid scheme 3 and temperature differences between the simulations using uniform and non-uniform grid.



(a) Evolution of the error.

(b) Evolution of α value.

Figure 4.31: Evolution of the error and the α value during the full-inverse process applied to $V82$ object (reprinted from (López, 2003).

Chapter 5

Conclusions and future work

In this work we have introduced the problem of non-destructive evaluation of the soil applied to the detection of buried landmines. There are several techniques to perform the landmine detection, however the use of infrared thermography has been presented as a good choice. The most challenging task in mine clearance is the detection of antipersonnel mines. These kind of mines is usually located near the surface. In this work we have focused on the use of infrared thermography as the scope applicability of this technique is 10-15 cm, where the antipersonnel are usually located. This technique must be used together with other techniques, such as ground penetrating radar, to detect mines located at higher depths. However, as we will focus on the detection of antipersonnel landmines this is a valid approach.

This thesis is based in previous works, where a landmine detection system was proposed. This system is based on the different thermal properties of mine and soil, these differences cause alterations in the thermal behavior of the soil, which is reflected in the IR images. The detection object system is based on a 3D thermal model of the soil which simulates its thermal behavior. This model is run several times in an inverse engineering process to obtain the location and nature of objects present in the soil. This is a very-high time-consuming process in ordinary computers.

The main objective of our work was to reduce the computing time, developing a portable system that could be used for on-field applications. To this aim we have worked in two issues:

- Implementation of a hardware FPGA-based system to solve the equations of the

thermal model of the soil using the FD-TD method.

- Reduction of the number of layers of the thermal model using non-uniform grids.

The main contribution of this work was the development of a FPGA implementation of the thermal model. The FPGA implementation acts as a hardware accelerator needed to reduced the excessive computing time. The validity of the synthesized architecture as a hardware accelerator was demonstrated comparing results obtained with MATLAB and C++ simulation with those given by the FPGA, obtaining a significant reduction in the computing time of up to 34 times compared to a PC-based solution (single precision on a Pentium IV 3GHZ). The speedup of the system is limited by the number of temperatures that can be simultaneously read from the memory banks. Taking into account the number of available $IOBS$ in the current FPGA, eight new memory banks could be added allowing to achieve an estimated speedup of 80 compared to the PC-based solution. We also estimated that a speedup of 210 could be achieved with more powerful FPGAs (Virtex-5). This demonstrates the validity of our implementation and it could be integrated into a portable system to be used for on-field applications. Additionally, the use of non-uniform grids is justified because from 20 cm the temperature of the soil varies very slowly and, on the other hand, the limit for the IRT is around 10-15 cm. Therefore, for deeper layers of 15 cm the separation between them can be increased, therefore the spatial resolution is reduced, allowing a reduction of the computing time in a 30% without introducing significant errors in temperatures at the surface that could affect the performance of the detection algorithm.

As future work we must improve the memory access, the bottleneck of the system. One way to improve this issue is the introduction of input buffers on the FPGA PEs for the memory banks 0 and 7. Fig. 3.15 showed the clock cycles necessary to read all neighbor temperatures in the temperature updating process of nodes *(i,j,k)*, *(i+1,j,k)*... *(i,j,k+7)*, *(i+1,j,k+7)*. As can be noted, in the 3rd read cycle only $PE0$ and $PE7$ receive data from $MBank7$ and $MBank0$, respectively, and the rest of the memory banks remain unused. To avoid this situation two buffers of size N_{x_M} (maximum numbers of points in the x direction) can be created. When surface nodes are being updated the reading scheme is the same that we have in the current implementation. However, with the proposed modification the values read in $MBank7$ ($T^m_{1,j,7}$ and $T^m_{i+1,j,7}$) and $MBank0$ ($T^m_{i,j,8}$ and $T^m_{i+1,j,8}$) could be stored in the corresponding buffer. Thus, once the complete sequence of nodes $(i,j,0)...(i,j,7)\,\forall i,j$ had been updated, the system would update the following sequence, i.e., $(i,j,8)...(i,j,15)\,\forall i,j$. In this case, the temperature values $T_{i,j,8}$ and $T_{i,j,7}$ (north neighbor) would be stored in

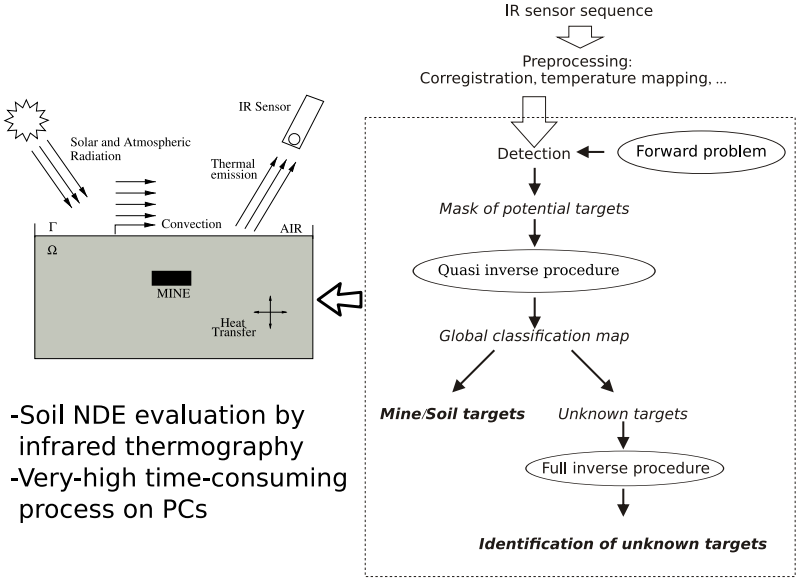
the buffers. Therefore, in $MBank0$ the value $T_{i,j,16}$ would be read, which is the south temperature neighbor of node *(i,j,15)*. Thus, in the first read cycle all south and north neighbors of all nodes are available, removing the 3rd read cycle of Fig. 3.15. Now, the buffers could store the values $T_{i,j,7}$ and $T_{i,j,8}$ $\forall i, j$ and the process is repeated. Using these two first in first out (FIFO) buffers, one clock cycle would be removed in the reading process. Thus a reduction of 25 % in the processing in the reading time is expected, which could be reflected in a reduction of 1 clock cycle in the updating process. However, the main limitation of this scheme is the size of the buffers that does not fit in the current FPGA, but that could fit in a Virtex-4 or Virtex-5.

Another line for future work is the implementation of the non-uniform grids on the FPGA. To introduce the variation in the F_0 parameters, some modifications of the PEs (add a multiplier and two adders), but they do not affect to the performance of the system as the update process would be performed using the same number of clock cycles.

Finally, a long-term work is the development of an FPGA board, where we could integrate several memory banks and maintaining the required programmability of the system.

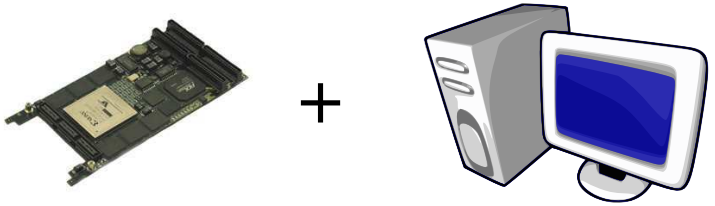
Hardware-based infrared non destructive evaluation for landmine detection

SOLUTION



-Soil NDE evaluation by infrared thermography
-Very-high time-consuming process on PCs

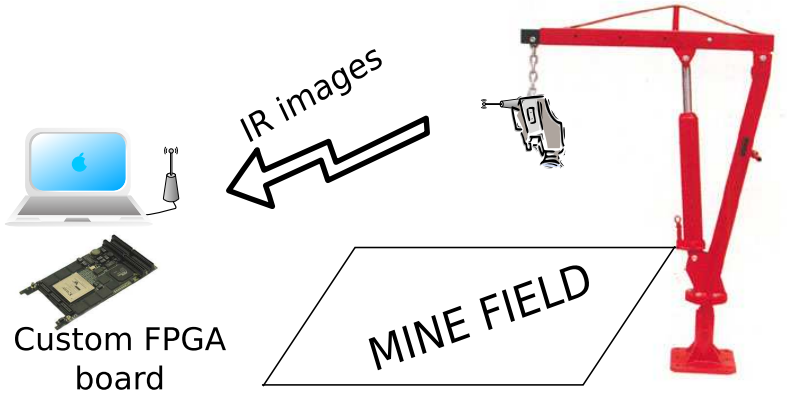
CURRENT IMPLEMENTATION



-Field Programmable Gate Array (FPGA)
-FPGA acting as an accelerator card
-Speeds up the computations (x30)

Material cost: FPGA (6000 €), PC (1000€), IR camera (9000€),
Software for FPGA design (1800€)
Personnel costs: 20000 €/year
Development time: 3 years
Total cost: 80000 €

FUTURE IMPLEMENTATION



Custom FPGA board

COST AND TIME ESTIMATION

Material: Virtex-5 (500 €) , board (200€), memory banks (300€) Laptop (1000€), wireless emisors (100€), IR camera (9000 €), crane (1000€)
Personnel costs: 20000€/year
FPGA board development development: 1 year
FPGA test: 3 months
System test on field applications: 3 months
Total cost: 40000 €

Index

AP mine, 34
AT mine, 34

Chen, 70
Chuan He, 73
CLB, 99
CNN, 75
computational tomography, 21
convection, 43
Culley, 72

DSP, 96
Durbano, 68

EC, 26
Electro-optical, 38
electromagnetic spectrum, 40
emissivity, 42

FD-TD, 8, 14, 50, 63, 94
forward problem, 14
FPGA, 15, 96

Gamma rays, 20
GPIO, 109
GPR, 13, 30, 70

heat equation, 13, 48

Infrared Thermography, 32
infrared thermography, 13
inverse problem, 14
IOB, 99
IRT, 37

landmines, 13
LPI, 22

Maxwell's equations, 64
MPI, 29
MRTD, 42

NDE, 13, 17
NDT, 17

PDE, 63
Placidi, 68
Planck's law, 40

resolution, 42

Schneider, 67
sensitivity, 42
solar radiation, 44
speed-up, 15
Stephan Boltzmann law, 47

thermal model, 14
thermal signature, 39

UT, 24

VHDL, 15, 68
Visual inspection, 20

Wien's displacement law, 41

X-rays, 20

Yee-Cell, 65

Glossary

AP	Antipersonnel, 34
AT	Antitank, 34
CLB	Configurable logic block, 99
CNN	Cellular neural/non-linear network, 74
CT	Computational tomography, 21
DSP	Digital signal processor, 96
DT-CNN	Discrete time CNN, 74
EC	Eddy current, 26
EMI	Electromagnetic induction, 28
EO	Electro-optical, 38
FD-TD	Finite-Difference Time-Domain, 8, 49
FOV	Field of view, 162
FPGA	Field programmable gate array, 14
FPU	Floating point unit, 67
GPIO	Global purpose input/output, 108
GPR	Ground penetrating radar, 13, 30
GPS	Global Positioning System, 36
HPC	High performance computing, 71
IOB	Input/Output block, 99

IRT	Infrared Thermography, 32
LPI	Liquid penetrant inspection, 22
MPI	Magnetic particle inspection, 29
MRTD	Minimum resolvable temperature difference, 42
NDE	Non destructive evaluation, 13
NDT	Non destructive testing, 17
PDE	Partial differential equation, 63
RTL	Register transfer level, 67
UT	Ultrasonic testing, 24
VHDL	Very high speed integrated circuit hardware description language, 14
VI	Visual inspection, 20
ZBT	Zero bus turnaround, 98

List of Symbols

H	local insolation function, 45
S_0	Solar constant, 45
T	Temperature, 40
α	Thermal diffusivity, 48
δ	latitude, 46
ϵ	emissivity, 42
λ	wavelength, 40
$\overrightarrow{B}$	magnetic field, 64
$\overrightarrow{D}$	electric displacement field, 64
$\overrightarrow{E}$	electric field, 64
$\overrightarrow{H}$	magnetic field strength, 64
$\overrightarrow{J}$	free current density, 64
ϕ	latitude, 46
ρ	density, 48
ρ_v	free electric charge density, 64
σ	Stephan-Boltzmann constant, 41
θ	solar zenith angle, 45
c_p	Specific heat, 48
k	Thermal conductivity, 48

Bibliography

(1997). Convention on the prohibition of the use, stockpiling, production and transfer of anti-personnel mines and on their destruction.

Ala-Laurinaho, J., Hirvonen, T., Tuovinen, J., and Räisänen, A. V. (1997). Numerical modeling of a nonuniform grating with fdtd. *Microwave and Optical Technology letters*, 15(3):134–139.

Anderson, N. and Sousa, C. (1995). Social cost of land mines in four countries: Afghanistan, bosnia, cambodia, and mozambique. *British Medical Journal*, (311):718–721.

Arpaci, V., Selamet, A., and Kao, S. (2000). *Introduction to Heat Transfer*. Prentice-Hall.

Azevedo, S., Gavel, D., Mast, J., and Warhus, J. (1995). Statement of capabilities: Micropower Impulse Radar (MIR) technology applied to mine detection and imaging. Technical report, Lawrence Livermore National Laboratory.

Bach, P., Toumeur, P. L., Poumarkde, B., and Bretteand, M. (1996). Neutron activation and analysis. In *EUREL International Conference Detection of Abandoned Landmines*, volume 431, pages 58–61.

Baldev, R., Jayakumar, T., and Thavasimuthu, M. (2002). *Practical Non-destructive Testing*. Woodhead Publishing.

Barret, H. . and Swindel, W. (1981). *Radiological Imaging: Theory of Image Formation, Detection and Processing*. Academic Press.

Beaven, S., Stockera, A., and Winter, E. (2004). Joint multisensor exploitation for mine detection. In *Proceedings of the SPIE*, volume 5415, pages 1094–1105.

Bejan, A. (1993). *Heat Transfe*. John Wiley & Sons, Inc.

Belanović, P. and Leeser, M. (2002). A library of parameterized floating point modules and their use. In *Proceedings of the International Conference on Field-Programmable Logic and Applications (FPL)*, pages 657–666.

Birnbaum and Free, editors (1981). *Eddy Current Characterization of Materials and Structures*. ASTM International.

Brankovic, V., Krupezevic, D., and Arndt, F. (1992). An efficient two-dimensional graded mesh finite-difference time-domain algorithm for shielded or open waveguide. *IEEE Transactions on Microwave Theory and Techniques*, 40(12):2272–2277.

Brunzell, H. (1998). *Signal Processing Techniques for Detection of Buried Landmines using Ground Penetrating Radar*. PhD thesis, Chalmers University of Technology.

Cameron, M. and Lawson, R. (1998). *To Walk Without Fear: The Global Movement to Ban Landmines*. Toronto: Oxford University Press.

Celoxica (2005a). *Handel-C Language Reference Manual for DK4*.

Celoxica (2005b). *RC2000 ADM-XRC-II Manual*. Celoxica.

Celoxica (2006). `http://www.celoxica.com`.

Chen, W., Kosmas, P., Leeser, M., and Rappaport, C. (2004). An FPGA implementation of the two-dimensional finite-difference time-domain (FDTD) algorithm. In *FPGA '04: Proceedings of the 2004 ACM/SIGDA 12th international symposium on Field programmable gate arrays*, pages 213–222. ACM Press.

Chen, W., leeser, M., and Rappaport, C. (2006). Acceleration of the 3D FDTD algorithm in fixed-point arithmetic using reconfigurable hardware. In *Progress in electromagnetics research symposium*, page 550.

Chua, L. and Yang, L. (1988). Cellular neural networks: theory and applications. *IEEE Transactions on circuits and Systems*, (35):1257–1290.

Conyers, L. B. and Goodman, D. (1997). *Ground-Penetrating Radar: An Introduction for Archaeologists*. Altamira Press.

Corning, D. (1981). Product information for silastic 3110 rtv silicone rubber. Technical Report 61-143-01.

Cremer, F., Schutte, K., Schavemaker, J., and den Breejen, E. (2001). A comparison of decision-level sensor-fusion methods for anti-personnel landmine detection. *Information Fusion*, 2:187–208.

Culley, R., Desai, A., Gandhi, S., Wu, S., Tomko, K., Sotirelis, P., and Richie, D. (2005). A prototype FPGA Finite-Difference Time-Domain engine for electromagnetics simulation. In *Proceedings of the 48th IEEE Int Midwest Symposium on Circuits & Systems*, volume 1, pages 663–666.

Curt, P. F., Durbano, J. P., Bodnar, M. R., Shi, S., and Mirotzni, M. S. (2007). Enhanced functionality for hardware-based FDTD accelerators. *Applied Computational Electromagnetics Society Journal*, 22(1):39–46.

D'Agostino, J. and Scott, L. (1993). Thermal imaging systems performance model. Technical Report 5008993, U. S Army Nigth Vision and Electronic Sensors Directorate.

Daniels, D. (1996). *Surface penetrating radar*. IEE Radar, Sonar, Navigation and Avionics.

de Jong, W., Lensen, H. A., and Janssen, H. L. (1999). Sophisticated test facilities to detect land mines. *Detection and remediation technologies for mines and minelike targets IV*, 3710:1409–1418.

de Jong, W. and Sjokvist, S. (2002). Tno-fel/foi multi spectral mine detection trial. Technical report, TNO Physics and Electronics Laboratory: FOI Swedish Defense Research Agency.

de Rivas, E. K. (1972). On the use of nonuniform grids in finite-difference equations. *J. Comput. Phys.*, 10:202–210.

Dee, W. M. (1998). *Analysis of transport phenomena*. New York Oxford University.

Dey, S., Mittra, R., and Chebolu, S. (1997). A technique for implementing the FDTD algorithm on a non orthogonal grid. *Microwave and optical technology letters*, 14(4):213–215.

Driggers, R., Cox, P., and Edwards, T. (1999). *Introduction to Infrared and Electro-Optical Systems*. Artech House.

Durbano, J. P., Ortiz, F. E., Humphrey, J. R., Mirotznik, M. S., and Prather, D. W. (2003a). Hardware implementation of a three-dimensional finite-difference time-domain algorithm. *IEEE Antennas and Wireless Propagation Letters*, 2(1):54–57.

Durbano, J. P., Ortiz, F. E., Humphrey, J. R., Prather, D. W., and Mirotznik, M. S. (2003b). Implementation of three-dimensional fpga-based fdtd solvers: an architectural overview. In *11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM03)*, pages 269–270.

EMPhotonics (2007). `http://www.emphotonics.com`.

Engl, H. W., Hanke, M., and Neubauer, A. (1996). *Regularization of Inverse problems*. Kluwer Academic Publishers.

England, A., Galantowiz, J., and Schretter, M. (1992). The radiobrightness thermal inertia measure of soil moisture. *IEEE Transactions on Geoscience and Remote Sensing*, 30(1):132–139.

England, A. W. (1990). Radiobrigthness of diurnally heated, freezing soil. *IEEE Transactions on Geoscience and Remote Sensing*, 28(4):464–476.

Englelbeen, A. (1998). Nuclear quadrupole resonance mine detection. In *CLAWAR'98*, pages 249–253.

Falconer, M. and Tripathi, V. (1997). Second order nonuniform grid spacing in the fdtd technique. In *Microwave Symposium Digest*, volume 3, pages 1539–1542. IEEE MTT-S International.

Grande, N. D., Durbin, P. F., and Perkin, D. E. (1997). Dual-band infrared computed tomography for quantifying aircraft corrosion damage. In *First Joint DOD/FAA/NASA conference on aging aircraft*.

Gray, S. H. (2001). Y2K review article: seismic migration problems and solutions. *Geophysics*, (66):1622–1640.

Gros, B. and Bruschini, C. (1998). A survey on sensor technology for landmine detection. *Journal of Humanitarian Demining*, pages 172–187.

Gschwind, M. and Salapura, V. (1995). A vhdl design methodolgy for fpgas. In *Proceedings of the 5th International Workshop on Field-Programmable Logic and Applications*, pages 208–217.

Hartley, R. and Parhi, K. K. (1995). *Digit-Serial computation*. Kuwler Academic Publishers.

He, C., Qin, G., Lu, M., and Zhao, W. (2006). Optimized high-order finite difference wave equations modeling on reconfigurable computing platform. *Microprocessors and Microsystems*, 31:103–115.

He, C., Zhao, W., and Lu, M. (2005). FPGA-based high-order finite difference algorithm for 2D acustic wave propagation problems. In *ERSA05*.

Hellier, C. (2000). *Handbook of Nondestructive Evaluation*. McGraw-Hill Education.

Hoffmann, J., Sathish, S., and Meyendorf, N. (2001). Evaluation of corrosion on aluminum alloy airframe structures under protective coatings using acoustic and thermographic techniques. In *In Proceedings of the 2nd International Conference on Light Materials for Transportation Systems*.

Horowitz, P. (1996). New technological approaches to humanitarian demining. Technical Report JSR-96-115, JASON MITRE.

ICBL (2006). *Landmine Monitor Report 2006*. International campaing to can landmines (ICBL).

ICBL (2008). International campaing to ban landmines (ICBL). `http://www.icbl.org/`.

Incropera, F. P. and DeWitt, D. P. (2002). *Introduction to Heat Transfer*. John Wiley & Sons, Inc.

Initiative, O. S. (2003). *SystemC 2.0.1 Language Reference Manual*.

Jankowski, P., Mercado, A., and Hallowell, S. (1992). FAA explosive vapor/particle detection technology. In *Applications of Signal and Image Processing in Explosives Detection Systems*, volume 1824, pages 13–27.

Ji, Z., Hagness, C., Booske, H., Mathur, S., and Meltz, M. (2006). FDTD analysis of a gigahertz TEM cell for ultra-wideband pulse exposure studies of biological specimens. *IEEE Transactions on Biomedical Engineering*, 53(5):780–789.

Kahle, A. B. (1977). A simple thermal model of the earth's surface for geologic mapping by remote sensing. *Journal of Geophysical Research*, 82:1673–1680.

Kak, A. and Slaney, M. (1987). *Principles of Computerized Tomographic Imaging*.

Kermani, M. and Ramahi, O. (2006). The complementary derivatives method: a second-order accurate interpolation scheme for non-uniform grid in fdtd simulation. *IEEE Microwave and wireless componets letters*, 16(2):60–62.

Khanafer, K. and Vafai, K. (2002). Thermal analysis of buried land mines over a diurnal cycle. *IEEE Transactions on Geoscience and Remote Sensing*, 40(2):461–473.

Kirsch, A. (1996). *An introduction to the Mathematical Theory of Inverse problems*, volume 120 of *Applied mathematical sciences*. Springer-Verlag, New York.

Knoll, M., Haeni, F., and Knight, R. (1991). Characterization of a sand and gravel aquifer using ground penetrating radar. Technical Report 91-4035, U.S. Geological survey Water Resources Investigations Report.

Kosmas, P., Wang, Y., and Rappaport, C. M. (2002). Three-dimensional FDTD model for GPR detection of objects buried in realistic dispersive soil. In *Proc. SPIE Vol. 4742, Detection and Remediation Technologies for Mines and Minelike Targets VII.*, pages 330–338.

Krautkrämer, J. and Krautkrämer, H. (1990). *Ultrasonic Testing of Materials*. Springer Verlag.

L. van Kempen, H. Sahli, J. B. and Cornelis, J. (2000). New results on clutter reduction and parameter estimation for landmine detection using GPR. In *Proc. GPR 2000, 8th International Conference on Ground Penetrating Radar*, pages 872–879.

Lanuzza, M., Margalla, M., and Corsonello, P. (2005). Low-cost fully reconfigurable data-path for FPGA-based multimedia processor. In *Proceedings of the 15th International Conference on Field Programmable Logic and Applications (FPL)*, pages 13–18.

Larsson, C. and Abrahamsson, S. (1993). Radar, multispectral and biosensor techniques for mine detection. In *Symposium on Anti-Personnel Mines*, pages 179–202.

Levine, Z., Kalukin, A., Frigo, S., I, I. M., and Kuhn, M. (1999). Tomographic reconstruction of an integrated circuit interconnect. *Applied Physics Letters*, 74(1):150–152.

Liou, Y. and England, A. (1998). A land surface process/radiobrightness model with couple heat and moisture transport for freezing soils. *IEEE Transactions on Geoscience and Remote Sensing*, 36(2):669–677.

Lockwood, G., Shope, S., Bishop, L., Selph, M., and Jojola, J. (1997). Mine detection using backscatered x-ray imaging of antitank and antipersonnel mines. *Detection and Remediation Technologies for Mines and Minelike Targets II*, 3079:408–417.

López, P. (2003). *Detection of Landmines from Measured Infrared Images using Thermal Modeling of the Soil*. PhD thesis, Universidad de Santiago de Compostela.

López, P., Pardo, F., Sahli, H., and Cabello, D. (2008). Non-destructive soil inspection using an efficient 3D software-hardware heat equation solver. *Inverse Problems in Science and Engineering*. (Invited paper, in evaluation).

López, P., van Kempen, L., Sahli, H., and Cabello, D. (2004). Improve thermal analysis of buried landmines. *IEEE Transactions on Geoscience and Remote Sensing*, 42(9):1965–1975.

Lovejoy, D. (1993). *Magnetic Particle Inspection: A practical guide*. Kluwer Academic Publishers.

Maksymonko, G., Ware, B., and Poole, D. (1995). A characterization of diurnal and enviromental effects on mines and the factors influencing the performance of mine detecting ATR algortihms. In *Detection and Remediation Technologies for Mines and Minelike Targets, Proceedings of th SPIE*, volume 2496, pages 140–151.

Maksymonko, G. B. and Le, N. (1999). Performance comparison of standoff minefield detection algorithms using thermal IR image data. volume 3710, pages 852–863. SPIE.

Maldague, X. P., editor (1994). *Infrared Methodology and Technology*. CRC Press.

Marchuk, G. I. (1975). *Methods of numerical mathematics*. Springer-Verlag, New York.

Marek, J. R., Mehalic, M. A., and Terzuoli, A. J. (1992). A dedicated VLSI architecture for finite-difference time domain calculations. In *8th Annual Review of Progress in Applied Computational Electromagnetics*, volume 1, pages 546–553.

McIntire, P. and Moore, P. (1996). *Visual and optical testing*, volume 8. ASNT.

Motuk, E., Woods, R., and Bilbao, S. (2005). Implementation of finite difference schemes for the wave equation on FPGA. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages 237–240.

Muscio, A. and Corticelli, M. A. (2004). Land mine detection by infrared thermography: reduction of size and duration of the experiments. *IEEE Transactions on Geoscience and Remote Sensing*, 42(9):1955– 1964.

Myer, K. (2004). *Handbook of Materials Selection*. John Wiley & Sons.

Nagy, Z., Vörösházi, Z., and Szolgay, P. (2006). Emulated digital CNN-UM solution of partial differential equations. *International Journal of Circuit Theory and Applications*, (34):445–470.

Navarro, E., Sangary, N., and Litva, J. (1996). Some considerations on the accuracy of the nonuniform FDTD method and its application to waveguide analysis when combined with the perfectly matched layer technique. *IEEE Transactions on Microwave Theory and Techniques*, 44(7):1115–1123.

Noel, T. and Moore, P. (1999). *Liquid Penetrant Testing, Nondestructive Testing Handbook*, volume 2. American Society for Nondestructive Testing, Columbus.

Ortigosa, E. M., Cañas, A., Ros, E., Ortigosa, P. M., Mota, S., and Díaz, J. (2006). Hardware description of multi-layer perceptrons with different abstraction levels. *Microprocessors and Microsystems*, 30:435–444.

Pardo, F., López, P., Balsi, M., and Cabello, D. (2006). FPGA implementation of 3-D thermal model simulator. In *Proceedings of FPL'06*, pages 633–636.

Pardo, F., López, P., and Cabello, D. (2007). Soft-hard 3D FD-TD solver for non destructive evaluation. In *Proceedings of FPL'07*, pages 17–22.

Pardo, F., López, P., and Cabello, D. (2008a). DT-CNN emulator: 3D heat equation solver with applications on the non-destructive soil inspection. In *Proceedings of the 11th International Workshop on Cellular Neural Networks and their Applications*, pages 11–16.

Pardo, F., López, P., and Cabello, D. (2008b). FPGA-based hardware accelerator of the heat equation with applications on infrared thermography. In *Proceedings of the 19th IEEE International Conference Application-specific Systems, Architectures and Processors (ASAP)*, pages 185–190.

Pardo, F., López, P., Cabello, D., and Balsi, M. (2008c). Efficient software-hardware 3D heat equation solver with applications on the non-destructive evaluation of minefields. *Computers & Geosciences*. (in evaluation).

Paul, D., Pothecary, N., and Railton, C. (1994). Calculation of the dispersive characteristics of open dielectric structures by the finite-difference time-domain method. *IEEE Transactions on Microwave Theory*, 42(7):1207–1212.

Placidi, P., Verducci, L., Matrella, G., Roselli, L., and Ciampolini, P. (2002). A custom VLSI architecture for the solution of fdtd equations. *IEICE Trans. Electron.*, E85(C):572–577.

Potter, M. C. and Wiggert, D. C. (2002). *Mechanics of fluids*. Pacific Grove CA: Brooks/Cole, 3rd edition.

Pramod, K. and Inaudi, D., editors (2000). *Trends in optical nondestructive testing and inspection*. Elsevier.

Pregowski, P., Walczack, W., and Lamorski, K. (2000). Buried mine and soil temperature prediction by numerical model. In *Proceedings of the SPIE, Detection and Remediation Technologies for Mines and Minelike Targets V*, volume 4038, pages 1392–1403.

Pruitt, D. and Lebsack, S. (1994). Return on investment from nde and maintenance expenses - a case history. *Materials evaluation*, 52(11):1279–1281.

Ratches, J. A., Vollmerhausen, R. H., and Driggers, R. G. (2001). Target acquisition performance modeling of infrared imaging systems: Past, present, and future. *IEEE Sensors Journal*, 1(1):31–40.

Rohsenow, W. and Hartnett, J. P. (1973). *Handbook of Heat Transfer*. McGraw-Hill.

Roy, S. and Banerjee, P. (2005). An algorithm for trading off quantization error with hadrware resources for matlab-based FPGA design. *IEEE Transactions on Computers*, 54(7):886–896.

Russell, K. L., McFee, J. E., and Sirovyak, W. (1997). Remote performance prediction for infrared imaging of buried mines. In *In Proceedings of SPIE 1996, Terrorism and Counterterrorism Methods and Technologies*, pages 762–769.

Sabatier, J. and Xiang, N. (2001). An investigation of a system that uses acoustic seismic coupling to detect buried anti-tank mines. *IEEE Transactions on Geoscience and Remote Sensing*, 39(6):1146–1154.

Samarskii, A. A. (2001). *The theory of difference schemes*. Pure and applied mathematics. CRC.

Savelyev, T. G., van Kempen, L., Sahli, H., Sachs, J., and Sato, M. (2007). Investigation of Time–Frequency features for GPR landmine discrimination. *IEEE Trans. Geosci. Remote Sensing.*, 45(1):118–129.

Schneider, R., Turner, L. E., and Okoniewski, M. (2002a). Application of FPGA technology to accelerate the Finite-Difference Time-Domain (FDTD) method. In *FPGA'02*, pages 97–105.

Schneider, R. N., Okoniewski, M., and Turner, L. (2002b). Custom hardware implementation of the finite-difference time-domain (FDTD) method. In *Microwave Symposium Digest, 2002 IEEE MTT-S International*, volume 2, pages 875–878.

Schneider, R. N., Okoniewski, M., and Turner, L. (2002c). Finite-difference time-domain method in custom hardware? *IEEE Microwave and Wireless Components Letters*, 12(12):488–490.

Schneider, R. N., Okoniewski, M., and Turner, L. (2004). A software-coupled 2D FDTD hardware accelerator. In *Antennas and Propagation Society International Symposium, 2004. IEEE*, volume 2, pages 1692–1695.

Sendur, I. and Baertlein, B. A. (2000). Numerical simulation of thermal signatures of buried mines over a diurnal ccyle. In *Detection and Remediation Technologies for Mines and Minelike Targets, Proceedings of the SPIE*, pages 156–167.

Siegel, R. (2002). Land mine detection. *IEEE Instrumentation & Measurement Magazine*, pages 22–28.

Sjokvist, S. (1999). *Heat transfer modelling and simulation in order to predict thermal signatures - The case of buried land mines.* Master thesis, Institute of Technology, Linkonpings Universeit.

Stratton, J. (1941). *Electromagnetic theory.* NewYork: McGraw-Hill.

Stutzman, W. L. and Thiele, G. A. (1997). *Antenna Theory and Design.* Wiley;.

Sullivan, C. and Saini, M. (2003). Software-compiled system design optimizes xilinx programmable systems. *Xcell Journal Online.*

Sundqvist, H. and Veronis, G. (1970). A simple finite-difference grid with nonconstant intervals. *Tellus*, 22(26-31).

Systems, F. (2005). `http://www.flirthermography.com/`.

Taflove, A. (1995). Reinventing electromagnetics: emerging applications for FD-TD computation. *IEEE Computational Science and Engineering*, 2(4):24–34.

Thanh, N. T., Hao, D. N., and Sahli, H. (2006). Thermal modelling for landmine detection: efficient numerical methods and soil parameter estimation. In *Detection and Remediation for mines and minelike targets XI*, volume 6217 of *SPIE*, pages 198–208.

Thanh, N. T., Sahli, H., and Hao, D. N. (2007). Finite-difference methods and validity of a thermal model for landmine detection with soil property estimation. *IEEE Transactions on Geoscience and Remote Sensing*, 45(3):656–674.

Verducci, L., Placidi, P., Ciampolini, P., Scorzoni, A., and Roselli, L. (2003). A standard cell hardware implementation for finite-difference time domain (FDTD) calculation. In *IEEE MTT-S International Microwave Symposium Digest*, volume 3, pages 2085–2088.

Verkhogliad, A., Isaev, A., Kuropyatnik, I., Mamontov, P., Moiseev, V., and Poteev, D. (2005). Application of IR thermography for defects detection in train car wheel pairs. In *Proceeding of Automation, Control, and Applications*, pages 483–495.

VHDL (1987). *IEEE Standard VHDL reference manual*. IEEE Standard 1076.

Vines, A. and Thompson, H. (1999). Thompson, beyond the landmine ban: Eradicating a lethal legacy. Technical report, Research Institute for the Study of Conflict and Terrorism.

Watson, K. (1973). Geologic applications of thermal infrared images. *Proceeeding of the IEEE*, 63(1):128–137.

Wolfe, W. and Zissis, G. (1978). *Infrared handbook*. Environmental Research Institute of Michigan.

Xilinx (2004). *Virtex-II Platform FPGAs: Complete Data Sheet*. Xilinx.

Xilinx (2006). Virtex -5: Complete data sheet. Technical report, Xilinx.

Yee, K. (1966). Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media. *IEEE Trans. Antennas and Propagation*, 16:302–307.

Zoubir, A. M. and Iskander, R. (2004). *Bootstrap Techniques for Signal Processing*. Cambridge University Press.

Zscherpel, U. and Alekseychuk, O. (2003). Crack detection in digital radiographs of weld inspection. In *2nd Workshop NDT in Progress*.