DIGITAL READOUT INTEGRATED CIRCUIT (DROIC) IMPLEMENTING TIME DELAY and INTEGRATION (TDI) for SCANNING TYPE INFRARED FOCAL PLANE ARRAYS (IRFPAs)

by

ÖMER CEYLAN

Submitted to the Graduate School of Engineering and Natural Sciences

in partial fulfillment of

the requirements for degree of

Doctor of Philosophy

Sabancı University

August 2016

DIGITAL READOUT INTEGRATED CIRCUIT (DROIC) IMPLEMENTING TIME DELAY and INTEGRATION (TDI) for SCANNING TYPE INFRARED FOCAL PLANE ARRAYS (IRFPAs)

APPROVED BY

Prof. Yaşar GÜRBÜZ (Thesis Advisor)	[illegible]………………………………………
Assoc. Prof. Ayhan BOZKURT	Ayhan Bozkurt………………………………………
Prof. Erkay SAVAŞ	[illegible]………………………………………
Assoc Prof. Arda Deniz YALÇINKAYA	[illegible]………………………………………
Assist. Prof. Erdinç ÖZTÜRK	[illegible]………………………………………

DATE OF APPROVAL: 10/08/2016

ABSTRACT

DIGITAL READOUT INTEGRATED CIRCUIT (DROIC) IMPLEMENTING TIME DELAY and INTEGRATION (TDI) for SCANNING TYPE INFRARED FOCAL PLANE ARRAYS (IRFPAs)

ÖMER CEYLAN

Ph.D. THESIS, AUGUST 2016

THESIS SUPERVISOR: PROF. DR. YAŞAR GÜRBÜZ

Keywords: Time Delay and Integration (TDI), Digital Readout Integrated Circuit (DROIC), Scanning Array Infrared Focal Plane Array (IRFPA), Digital TDI ROIC

This thesis presents a digital readout integrated circuit architecture implementing time delay and integration with an oversampling rate of 2 for scanning type infrared focal plane arrays with a charge handling capacity of 44.8 Me$^-$ while achieving quantization noise of 198 e$^-$ and power consumption of 14.5 mW. Conventional pulse frequency modulation method is supported by a single slope ramp ADC technique to have very low quantization noise together with a low power consumption. The proposed digital TDI ROIC converts the photocurrent into digital domain in two phases; in the first phase, most significant bits are generated by conventional PFM technique in the charge domain, while in the second phase least significant bits are generated by the single slope ramp ADC in the time domain. A 90x8 prototype has been fabricated and verified, showing a significantly improved signal-to-noise ratio of 51 dB for the low illumination levels (280,000 collected electrons), which is attributed to the TDI implementation method and very low quantization noise due to the single slope ADC. The proposed digital TDI ROIC proves the benefit of digital readouts for the scanning arrays enabling smaller pixel pitch, better SNR for low illumination levels and lower power consumption compared to the analog TDI readouts.

ÖZET

TARAMALI TİP KIZILÖTESİ GÖRÜNTÜLEME SİSTEMLERİ İÇİN ZAMAN ERTELEMELİ BİRİKTİRME (TDI) METODUNU UYGULAYAN SAYISAL TÜMLEŞİK OKUMA DEVRESİ

ÖMER CEYLAN

Ph.D. TEZİ, AĞUSTOS 2016

TEZ DANIŞMANI: PROF. DR. YAŞAR GÜRBÜZ

Anahtar Kelimeler: Zaman Ertelemeli Biriktirme (TDI), Sayısal Tümleşik Okuma Devresi (DROIC), Taramalı Tip Kızılötesi Görüntüleme Sistemi (IRFPA), Sayısal TDI ROIC

Bu tez taramalı tip kızılötesi görüntüleme sistemleri için zaman ertelemeli biriktirme uygulayan sayısal tümleşik okuma devresini sunmaktadır. Geliştirilen sayısal okuma devresi 14.5 mW güç tüketirken, 198 e^- niceleme gürültüsüne ve 44.8 Me^- yük depolama kapasitesine sahiptir. Literatürde bilinen darbe frekans modülasyonu tekniği, çok düşük niceleme gürültüsüne ulaşmak ve düş güç tüketmek için tek rampalı ADC metoduyla desteklenmiştir. Geliştirilen sayısal okuma devresi gelen fotoakımı iki bölümde sayısal veriye çevirmektedir. İlk bölümde en önemli bitler darbe frekans modülasyonu metoduyla oluşturulurken ikinci bölümde az önemli bitler tek rampalı ADC kullanılarak oluşturulmaktadır. Önerilen metodun doğrulaması 90x8 formatında prototip entegre devre üretilerek yapılmıştır. Yapılan ölçümlerde az önemli bitler oluşturulurken kullanılan tek rampalı ADC metodu ve TDI metodu sayesinde sinyal-gürültü-oranının düşük ışıma seviyelerinde 51 dB ile önemli derecede geliştirildiği görülmüştür. Geliştirilen sayısal TDI tümleşik okuma devresi, daha küçük piksel tasarımını mümkün kılması, düşük güç tüketimi ve özellikle düşük ışıma seviyelerinde elde ettiği yüksek SNR değerleri ile sayısal okuma devrelerinin taramalı tip kızılötesi görüntüleme sistemleri için de faydalı olduğunu göstermiştir.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my thesis supervisor Prof. Dr. Yaşar Gürbüz for the continuous support of my Ph.D. study and related research, for his patience, motivation, and immense knowledge. I have taken my courses from him, completed many projects successfully with him during my graduate study. His guidance helped me in all the time of research and writing of this thesis.

Besides my supervisor, I would like to thank my core thesis committee members: Assoc. Prof. Dr. Ayhan Bozkurt and Prof. Dr. Erkay Savaş, from whom I have taken many courses during my study at Sabancı University. I also would like to thank the rest of my thesis committee members: Assoc. Prof. Dr. Arda Deniz Yalçınkaya and Assist. Prof. Dr. Erdinç Öztürk for their insightful comments and encouragement.

My thesis work was supported by the Ministry of Science, Industry and Technology through the SANTEZ program with the project code 1245.STZ.2012-1. I would like to thank Prof. Dr. Uğur Çam for monitoring the SANTEZ project and for his comments on the thesis work also.

I thank my fellow labmates for the stimulating discussions, for the sleepless nights we were working together before tape-outs, and for all the fun we have had. I especially thank the ROIC group members Dr. Hüseyin Kayahan, Melik Yazıcı, Atia Shafique, Arman Galioğlu and Shahbaz Abbasi, with whom I have completed many projects. Melik Yazıcı deserves another thank for his financial support when needed. I also thank other lab members contributed to this thesis: Atia Shafique, Abdurrahman Burak, Can Çalışkan and Umut Çelik. Finally, I want to thank all members of IC Design and Test Lab: İlker Kalyoncu, Emre Özeren, Barbaros Çetindoğan, Berktuğ Üstündağ, Murat Davulcu, Eşref Türkmen, Elif Gül Özkan Arsoy, Dr. Geetesh Mishra, Divya Mahalingam and Ali Kasal. In particular, I want to thank Mehmet Doğan for his help on the drawing and fabrication of PCBs.

Last but not the least, I would like to thank my wife Nazlı and my daughter Fatma Zehra for supporting and motivating me to finish my Ph.D. I also would like to thank my parents Abdul Münip and Zeynep Ceylan, and to my brother Bilal, and my sisters Betül and Hilal for supporting me spiritually throughout my graduate study and my life in general.

TABLE of CONTENTS

LIST of FIGURES

LIST of TABLES

LIST of ABBREVIATIONS

ADC	Analog-to-Digital Converter
CCD	Charge Coupled Device
CDS	Correlated Double Sampling
CMOS	Complementary Metal-Oxide Semiconductor
CTAT	Complementary To Absolute Temperature
DAC	Digital-to-Analog Converter
DFF	D Flip-Flop
DI	Direct Injection
DPS	Digital Pixel Sensor
DROIC	Digital Readout Integrated Circuit
DUT	Device Under Test
EM	ElectroMagnetic
FLIR	Forward Looking InfraRed
FPA	Focal Plane Array
HDL	Hardware Description Language
I/O	Input/Output
IR	InfraRed
IRFPA	Infrared Focal Plane Array
IRST	InfraRed Search and Track
LSB	Least Significant Bit
LWIR	Long-Wave InfraRed
MBE	Molecular Beam Epitaxy
MEMS	MicroElectroMechanical System
MIM	Metal-Insulator-Metal
MOS	Metal-Oxide Semiconductor
MP	Mega Pixel
MSB	Most Significant Bit
MSO	Mixed Signal Oscilloscope

MUX	MUltipleXer
MWIR	Mid-Wave InfraRed
N/A	Not-Available
NMOS	N-type MOS
NIR	Near InfraRed
PFM	Pulse Frequency Modulation
PMOS	P-type MOS
PTAT	Proportional To Absolute Temperature
QDIP	Quantum Dot Infrared Photodetector
QWIP	Quantum Well Infrared Photodetector
ROIC	Readout Integrated Circuit
SCD	Semi-Conductor Devices
SLS	Strained Layer Superlattice
SNR	Signal-to-Noise Ratio
SPRITE	Signal Processing In The Element
SWaP	Size-Weight and Power
SWIR	Short-Wave InfraRed
TDI	Time Delay Integration
TRL	Technology Readiness Level
UAV	Unmanned Aerial Vehicle
US	United States
VIS	Visible
VLWIR	Very Long-Wave InfraRed
WWII	World War II

1. INTRODUCTION

1.1 History of the Infrared

Infrared radiation was first discovered by Herschel in 1800 by realizing that thermometer temperature rose 8 degrees in 16 minutes if the thermometer was centered ½ inch out of visible rays when the light goes through a prism [1]. The IR spectrum begins at 700 nm with Near IR and goes up to 20 µm with very longwave IR. However, atmospheric gasses like H_20, CO_2, O_2 limits the transmission of the IR radiation in some wavelengths. Figure 1 shows the atmospheric transmission together with IR spectrum. Due to these cut-offs in some wavelength intervals together with the wavelength itself, IR spectrum is divided into five main bands; namely near IR, shortwave IR, mid-wave IR, longwave IR and finally very long wave IR. Table 1 shows how IR bands are divided.

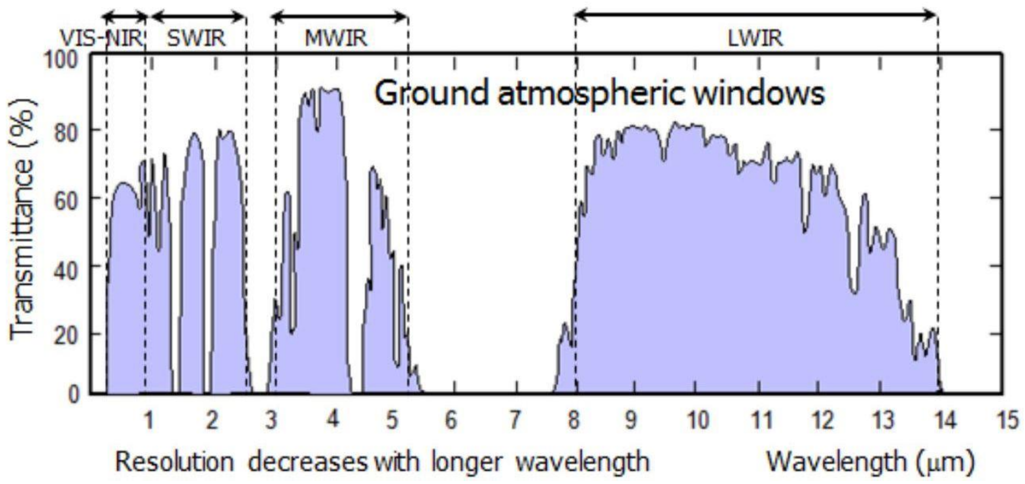


Figure 1: Atmospheric transmission and IR spectrum [2]

Table 1: IR bands and their wavelength intervals

Band	NIR	SWIR	MWIR	LWIR	VLWIR
Wavelength (µm)	0.7-1	1-2.5	3-6	8-14	15-20

Following the discovery of IR radiation, the first IR detector except the ordinary thermometer was discovered by Seebeck in 1821. He observed that a small electric current flows in a closed circuit of two dissimilar metallic conductors when their junctions are kept at different temperatures, which is called thermoelectric effect. Based on the thermoelectricity discovered by Seebeck, Nobili made the first thermocouple in 1829. Melloni improved the thermocouple by connecting multiple of them in series, which is called the thermopile and was 40 times more sensitive than a thermometer. In 1880, Langley invented the first bolometer, which he improved the sensitivity 400 times in the following 20 years, and was much more sensitive than the contemporary thermopiles [3]. Figure 2 shows a brief history of the infrared detectors. Although there were some attempts to develop photoconductive cells, detection of IR radiation was done with thermal detectors until 1940.

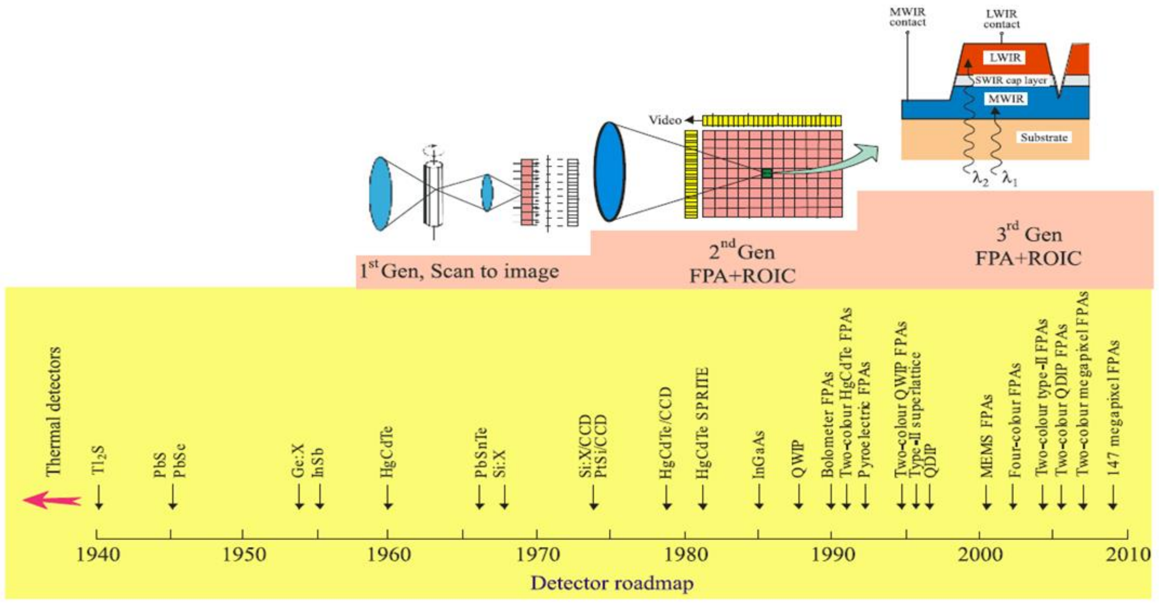


Figure 2: A brief history of infrared detectors [4]

The first photoconductive material that brought to manufacturing stage by Kutzscher in 1943 was lead sulphide (PbS). In the same years, Cashman developed the technology of thallous sulphide (Tl_2S) in 1941 and then concentrated his efforts on lead sulphide, which was produced at Northwestern University in 1944. The discovery of transistor in 1947 led to considerable improvement in IR detection technology also. In the early 1950s, the first high-performance extrinsic detectors were based on germanium using copper, mercury, zinc, gold as impurities led to many devices operating in the IR range of 8-14 μm. Mercury which is the most popular material for IR detection, came into the picture in 1960. At the beginning of the 1960s, narrow bandgap semiconductor alloys such as $InAs_{1-x}Sb_x$, $Pb_{1-x}Sn_xTe$ and $Hg_{1-x}Cd_xTe$ are introduced, which enabled customization of detectors for their spectral response. Among these alloys, the most popular one is $Hg_{1-x}Cd_xTe$ and it was patented in 1959 by Royal Radar Establishment. The popularity of HgCdTe detectors comes from its tunable alloys enabling a wide range of IR spectrum coverage. HgCdTe alloys can be tailored to detect incoming IR radiation between wavelength interval of 1 μm and 20 μm. Other advantages are high optical absorption coefficient, high electron mobility, and low thermal generation rate. At the beginning of the 1980s, there was an important invention called SPRITE (Signal Processing In The Element) that improves the signal-to-noise ratio of the scanning array systems. SPRITE enabled the integration of the signal multiple times with different pixels and averaging the final summed output signal, hence improving the SNR. Since it is difficult to grow HgCdTe material and high vapor pressure is required, alternative materials such as strained layer superlattices (SLSs), quantum-well infrared photodetectors (QWIPs) and quantum-dot infrared photodetectors (QDIPs) were investigated. First, in the 1980s, InGaAs was introduced, which has a similar performance with HgCdTe in the wavelength range of $1.5 < \lambda < 3.7$ μm due to its similar band structure. However, for mid-wave and long-wave infrared regions, InGaAs performance was poor due to substrate lattice mismatch. At end of the 1980s, QWIP were introduced for the LWIR region, which was thin layers of GaAs and AlGaAs. Due to the use of standard manufacturing techniques, highly uniform MBE growth, high yield, low cost, thermal stability, high impedance, fast response time, low power consumption and intrinsic radiation hardness, QWIPs gained attention. However, they cannot compete with HgCdTe detectors, since they have very low quantum efficiency, narrow spectral response and limited performance for short integration time applications. Another alternative detector material for the IR detection is the SLS that is generally composed of $InAs/Ga_{1-x}In_xSb$. SLSs are the potential candidate to replace HgCdTe with similar parameters like detectivity, impedance, optical gain, wide-band tunability. However, they are not mature enough to replace HgCdTe, since carrier lifetime is around 100 ns, problems exist in material growth, processing, substrate preparation and device passivation. Another alternative material is the QDIP, which has the bias-dependent spectral response and has the potential of being a smart sensor that can be tuned depending on the desired application. However, the absorption coefficient of QDIPs is not enough to have a satisfying performance [4].

Table 2: Comparison of device systems for LWIR spectral range [4]

Detector Type	**Bolometer**	**HgCdTe**	**Type-II SLs**	**QWIP**	**QDIP**
TRL Level	TRL 9	TRL 9	TRL 2-3	TRL 8	TRL 1-2
Status	Material choice for application requiring medium to low performance	Material choice for application requiring medium-high performance	Research and development	Commercial	Research and development
Military system examples	Weapon sight, night vision goggles, missile seekers, small UAVs, unattended ground sensors	Missile intercept, tactical ground, air born imaging, hyperspectral, missile seeker, missile tracking, space-based sensing	Being developed in university and evaluated in industry research environment	Being evaluated for some military applications	Very early stages of development at universities
Limitations	Low sensitivity and long time-constants	Performance susceptible to manufacturing variations. Difficult to extend to >14µ cutoff	Requires a significant >$100M, investment and fundamental material breakthrough to be mature	Narrow bandwidth and low sensitivity	Narrow bandwidth and low sensitivity
Advantages	Low cost and requires no active cooling, leverages standard Si manufacturing equipment	Near theoretical performance, will remain material of choice for minimum of the next 10-15 years	Theoretically better than HgCdTe at >14µ cutoff, leverages III-V fabrication techniques	Low-cost applications. Leverages commercial manufacturing processes. Very uniform material	Not sufficient to characterize material advantages

Table 2 shows the comparison of LWIR detector materials. For low to medium performance applications, bolometers are the optimum solution for cost effective decision. HgCdTe is the only choice for high-performance military applications for now, and it seems that it will be the main choice for a couple of decades. SLSs are good candidates to replace HgCdTe in the future, but they are still in research and development stage and need some time to be mature enough. QWIPs are widely used especially for the commercial applications and big array sizes with some limitations. Finally, QDIPs are at very early stage of research and development, and it seems that many years are required for this technology to reach technology readiness level (TRL) of 8-9.

1.2 Infrared Imaging Systems

Infrared imaging technology has been developed first in the 1950s to be used in military surveillance systems and it has been a very useful tool for many military and civilian applications such as IR search and track [5] [6], medical examination [7], astronomy [8], forward-looking infrared (FLIR) systems [9], missile guidance [10], security [11], process control [12], environmental monitoring [13] and spectroscopy [14]. The main motivation for the IR technology was military demands at the beginning, and it is still the main driving force to develop high-performance IR detectors and imaging systems. The first IR sensitive camera was developed by a Hungarian physicist Kalman in 1929 for anti-aircraft defense in Britain. The first IR line scanner was developed by the USA in 1947 after the WWII. The first real production FLIR based on Hg-doped Ge and using the 176-element array, was built for Air Force B52 in 1969 [4]. All these first products prove that military is the first user of the IR technology and keeps this technology advancing. However, after the 1990s with the cost reduction in the semiconductor industry, civilian applications have been gaining ground. Another reason that the civil applications come into the picture late is the secrecy of the advanced military technology, especially in the cold war times.

Figure 3 shows some civilian and military applications, that some of them mentioned previously. Today, IR technology is used in satellites for various civilian and military purposes ranging from agriculture, environmental monitoring, natural disaster mitigation to law enforcement and homeland security [15]. IR technology is also started to be used widely in consumer electronics such as in automotive industry for safe driving [16] and is integrated to smartphones [17] for everyday use of many people.

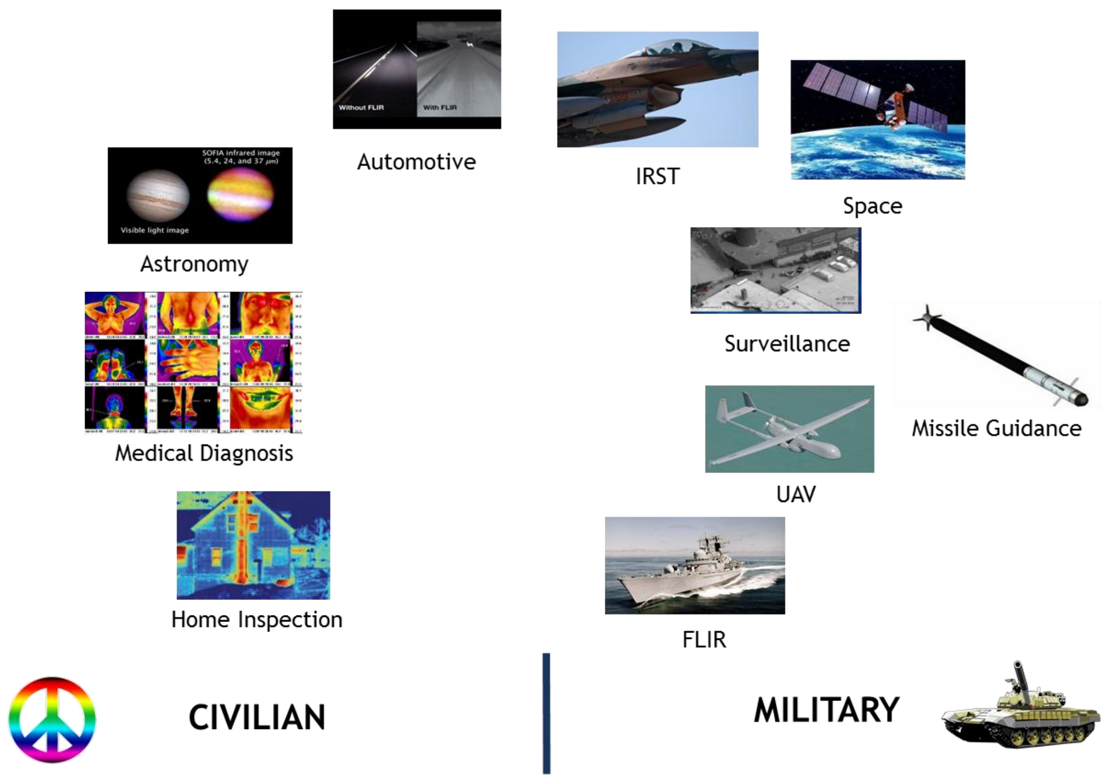


Figure 3: Civilian and military applications of IR technology

Table 3: IR imaging system applications *[18]*

COMMUNITY	APPLICATIONS	
MILITARY	Reconnaissance Target acquisition Fire control Navigation Missile guidance	
COMMERCIAL	CIVIL	Law enforcement Firefighting Border patrol
	ENVIRONMENTAL	Earth resources Pollution control Energy conservation
	INDUSTRIAL	Maintenance Manufacturing Nondestructive testing
	MEDICAL	Mammography Soft tissue injury Arterial construction Cancer detection

Table 3 shows the summary of civilian and military applications of the IR imaging technology. Apart from the applications previously mentioned, the IR imaging technology is also used in firefighting, maintenance of especially electrical systems, nondestructive testing of many devices such as LCD displays, arterial construction and cancer detection in civilian side. In the military side, reconnaissance applications, fire control systems, target acquisition systems and navigation systems also use the IR technology as one of their core modules. Performance requirements of the civilian and military applications differ significantly understandably. *Table 4* shows different requirements for civilian and military applications. Military applications require more sensitive, high resolution, vibration stabilized, real time and systems that can detect images from the long distance. On the other hand, civilian applications require moderate sensitivity and resolution together with cost effective systems as the market dictates.

Table 4: Design requirements for commercial and military applications *[18]*

DESIGN AREAS	MILITARY	COMMERCIAL
Vibration stabilized	Needed for long-range applications	Usually not required
Image processing algorithms	Application-specific (e.g., target detection or automatic target recognition)	Menu-driven multiple options
Resolution	High resolution (resolve targets at long distances)	Typically, not an issue because the image can be magnified by moving closer
Image processing time	Real-time	Real-time not always required
Target signature	Usually just perceptible	Usually high-contrast target
Sensitivity	Low noise (i.e., high sensitivity)	Noise not necessarily a dominant design factor (i.e., moderate sensitivity)

Figure 4 shows a general block diagram of an IR imaging system [19]. IR radiation is collected and focused to a detector with the specially designed optics for desired wavelength

interval. If the system is a scanning array system, there is a mechanical line scanner between the optics and detector. Detector is connected with the readout electronics (ROIC) which converts the photocurrent to voltage domain, amplify and multiplex the data coming from pixels and readout the final data either as an analog output or as a digital output. Finally, the output of the ROIC is connected to proximity electronics to be processed and sent to display.

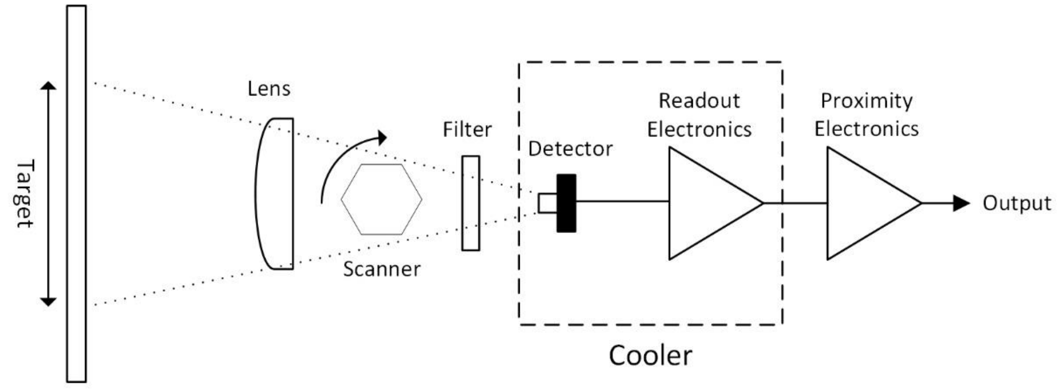


Figure 4: IR imaging system *[19]*

Figure 5 shows more detailed descriptions of the scanning array and staring array systems. Scanning array systems use single column or a couple of columns to generate a single column image. This single column generates the vertical part of the image while the horizontal part of the image is generated by the help of the scanner. First generation systems do not include the multiplexing functions in the focal plane array, hence each element of the column has an electrical contact outside the cryogenic cooler which makes these systems too bulky. First generation systems are not commercially available today. Second generation scanning array systems include the multiplexing functions in the focal plane array as part of the ROIC. Hence single or just a few cables are going out of the cryogenic cooler, which means less complex and costly dewars for the cryogenic cooling operation. There are many examples of the second generation systems on the market for both scanning and staring arrays. AIM's 288x6 and 576x7 arrays, Sofradir's 288x4 and 480x6 arrays are examples of the second generation scanning array systems. On the other hand, staring array systems have detector elements in a 2D array format, so they do not need a mechanical scanner to create a 2D image. Since multiplexing of staring array systems is much more complex than scanning array systems, all staring array systems are either second generation or third generation systems. Sofradir's 320x256 MWIR/LWIR and 640x512 MWIR/LWIR arrays with various pitch sizes, AIM's 384x288 and 1024x256 MWIR/SWIR arrays are examples of the second generation systems.

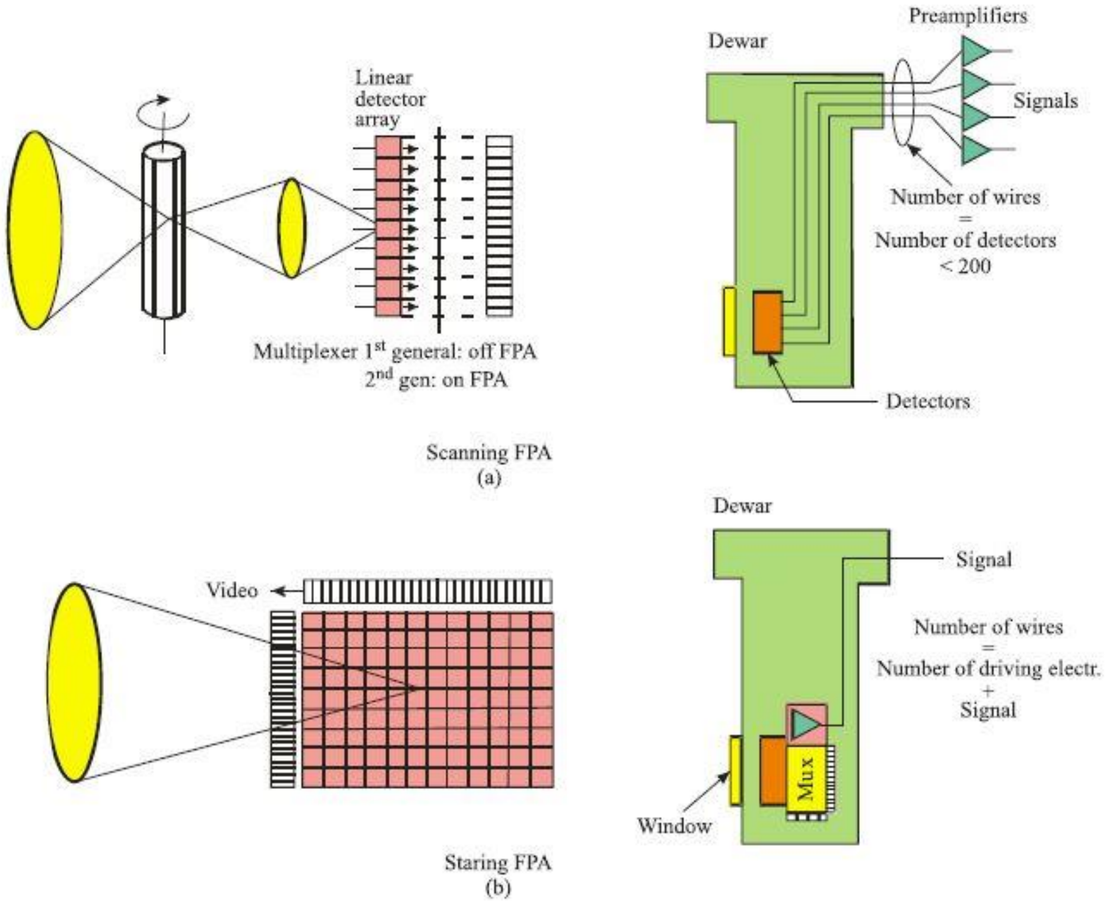

Linear detector array
Multiplexer 1st general: off FPA
2nd gen: on FPA
Scanning FPA
(a)
Dewar
Preamplifiers
Signals
Number of wires = Number of detectors < 200
Detectors
Video
Staring FPA
(b)
Dewar
Signal
Number of wires = Number of driving electr. + Signal
Mux
Window

Figure 5: Scanning array and staring array systems *[4]*

Finally, third generation systems were introduced which have a larger number of pixels, higher frame rates, better thermal resolution, multicolor functionality and other on-chip functions [3]. Raytheon [20], SCD [21], AIM [22], Sofradir [23] developed dual color FPAs for different IR bands. Cincinnati Electronics' ultra-high resolution 16 MP (4k x 4k) MWIR camera is also a good example of third generation large format FPAs [24]. Dual-color FPAs provide a very important capability to the user in terms of providing the accurate temperature of a target with unknown emissivity to identify the objects [25]. For example, LWIR provides the temperature information while SWIR provides the object shape and details, which results in an excellent collaboration different IR bands for target recognition. This feature of dual color FPAs is extensively used in satellites for Earth and planetary remote sensing and astronomy.

1.3 Readout Integrated Circuits (ROICs)

ROIC is one of the critical blocks of the IR imaging systems, it integrates the photocurrent coming from the detector, process it with low noise and outputs the final data pixel by pixel as an output. It performs TDI operation or correlated double sampling (CDS) to enhance the noise performance of the readout depending on the detector array format and application. An ideal IR imaging system should have detector limited noise performance, not ROIC limited noise performance, which makes the ROIC performance critical in terms of noise. The quality of the final image strongly depends on the ROIC performance. ROIC is also very important in terms of power consumption, frame rate, cooling cost, and FPA area. Depending on the application, ROICs are fabricated mostly in CMOS process and flip-chip bonded with detector substrates which are generally different materials due to the need for lower bandgap semiconductors for the detection of the desired wavelength. Some of the detector materials are HgCdTe, InGaAs, InSb, type-II strained layer superlattice detectors like InAs / GaInSb, quantum well infrared photodetectors (QWIPs) like AlGaAs / GaAs [26]. *Figure 6* shows flip-chip bonding technique for the integration of detector and ROIC. There are pad openings in ROIC pixels for the indium bumps which form the connection between the ROIC pixel array and the detector array. In this technique, ROIC and detector are fabricated and optimized separately and flip-chip bonded later with indium bumps [27].

ROICs can be divided into two categories in terms of signal processing method. The conventional method is the analog readout method which has been used since the 1970s and the second method is the digital readout method which has been gaining ground after 2000s. *Figure 7* show analog readout method which includes a unit cell containing input amplifier, a sample and hold circuit with integration capacitor; column and row multiplexers; bias and clock circuitry to control pixels, multiplexers and clock generation; column buffers to output the column data and an output driver that drives the output high capacitance load [28].

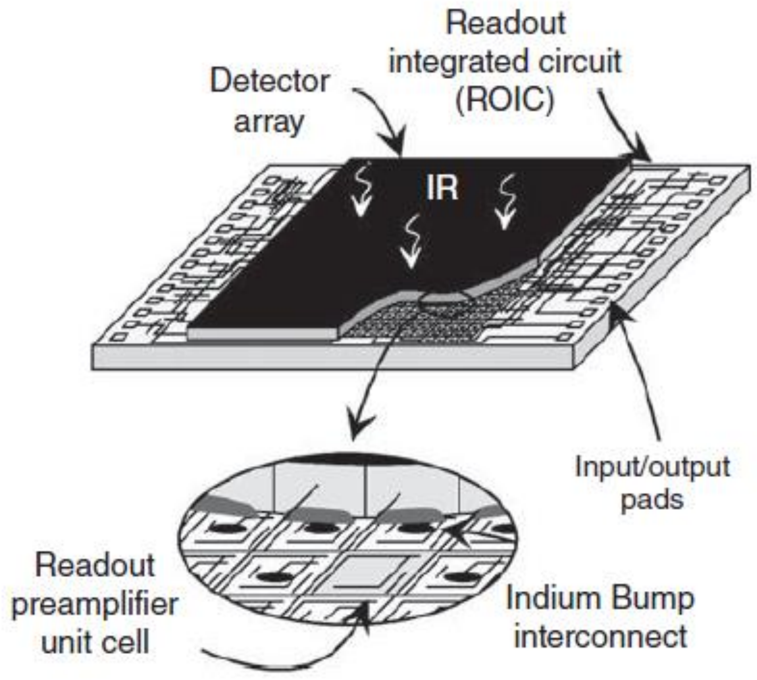

Readout integrated circuit (ROIC)
Detector array
IR
Input/output pads
Readout preamplifier unit cell
Indium Bump interconnect

Figure 6: Detector and ROIC hybridization with indium bumps *[27]*

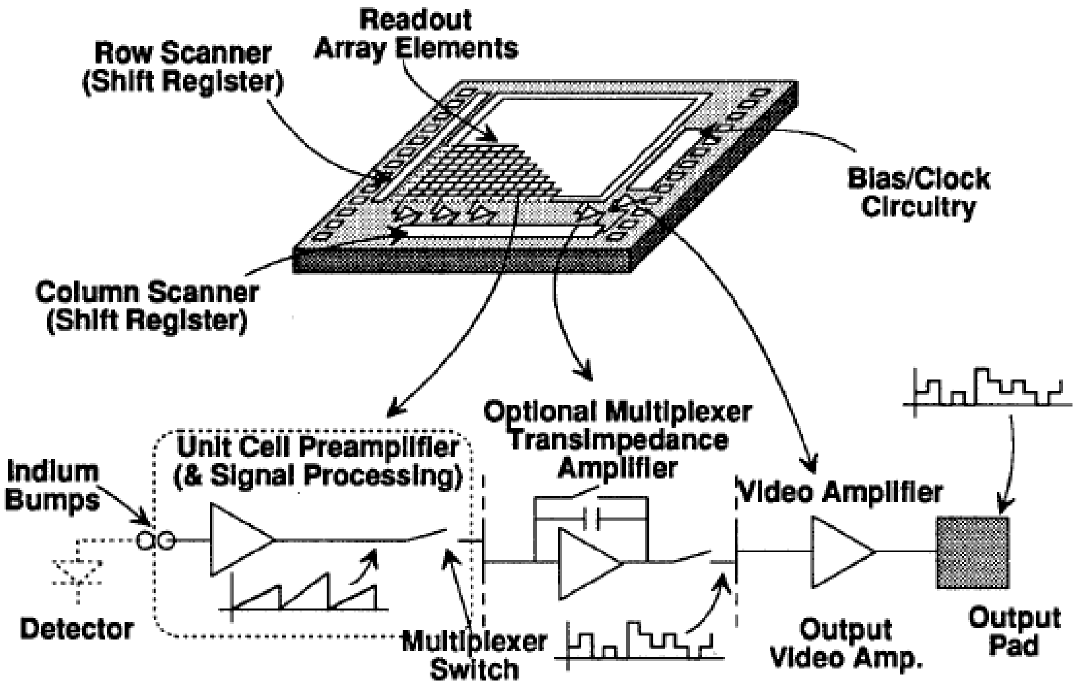

Readout Array Elements
Row Scanner (Shift Register)
Bias/Clock Circuitry
Column Scanner (Shift Register)
Optional Multiplexer Transimpedance Amplifier
Unit Cell Preamplifier (& Signal Processing)
Indium Bumps
Detector
Multiplexer Switch
Video Amplifier
Output Video Amp.
Output Pad

Figure 7: Conventional analog ROIC *[28]*

On the other hand, DROICs digitizes the data coming from photodetectors at different stages depending on the architecture. *Figure 8* shows three types of DROICs, first of which analog signal chain ending up with an on-chip serial ADC. The advantage of this ROIC is the on-chip analog to digital conversion which reduces the readout noise as opposed to the on-board AD conversion and it also makes it easier to integrate the ROIC/detector assembly into the imaging system. The main disadvantages of this method are extreme data rate bottleneck and high power consumption for large arrays. The second method for DROICs is the column parallel ADC method, which has an ADC per column. Each pixel in a column is multiplexed by the row multiplexer and connected to the column ADC. After the analog-to-digital conversion is done, outputs of each column ADC are also multiplexed to the output by a fast column multiplexer. Pixel output and signal chain in this method are still analog. Pixel output is multiplexed and buffered in the analog domain to the column-parallel ADCs. The main advantages of this method are lower power consumption and higher bandwidth compared to the first method. Main drawbacks of this method are area constraints to fit an ADC into a column and cross-talk from one column to another. The third DROIC is the pixel ADC method which involves an ADC in each single pixel. This method is also called pixel parallel or digital pixel sensor (DPS) architecture. Different than the first two DROIC methods, this method has digital pixel output and signal chain. The digital output of each pixel is multiplexed by row and column multiplexers and connected to the output. Benefits of this method are the highest bandwidth for large arrays and extreme charge handling capacity [29]. Depending on the circuit architecture used in the pixel to convert data from analog domain to digital domain, it is also possible to achieve low power consumption together with low quantization noise, hence improved SNR values. Drawbacks of this method are difficulty to fit into small pixel pitches and signal distribution issues. Especially, clock distribution to all pixels while trying to keep it away from sensitive analog lines is challenging.

There are various performance parameters related to the ROICs in order to satisfy the system requirements. As explained in Section 1.2, ROICs are one of the important sub-blocks of the IR imaging system and should not be the limiting component of the system. These performance parameters are noise, power consumption, input impedance, linearity, non-uniformity, pixel pitch, frame rate, SNR, charge handling capacity, gain and dynamic range.

Table 5 shows the performance parameters of the ROICs, their significance and how they are related to the system.

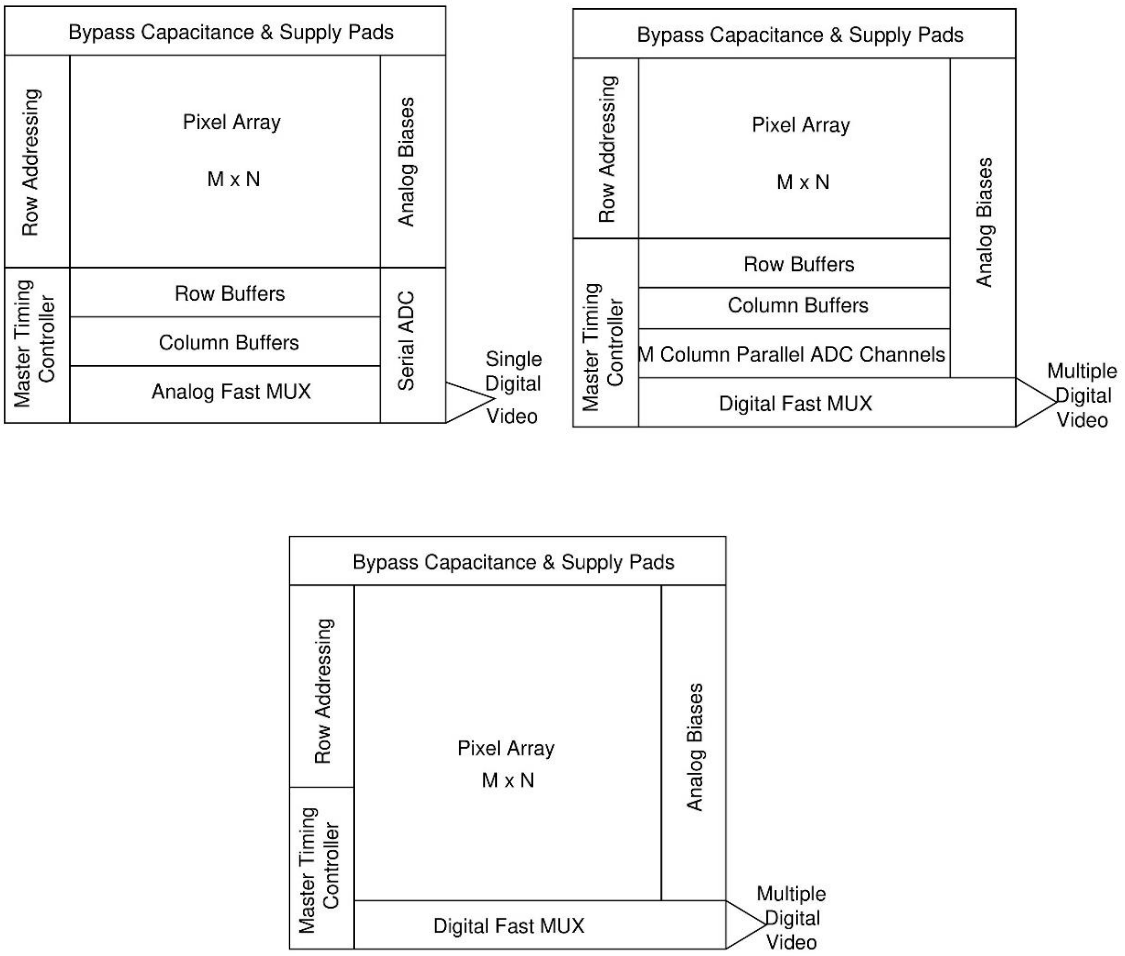


Figure 8: a) Serial ADC b) Column parallel ADC c) Pixel ADC for DROIC *[29]*

Table 5: Performance parameters of the ROICs

Performance Parameter	**Definition**	**Comments**
Noise	Variation of the measured signal. It can be temporal, spatial or both.	Determines minimum detectable signal, limiting factor for the sensitivity.
Power consumption	Power consumption of the ROIC at desired operating temperature and frame rate	Determines the cooling cost of the system. If the system is portable, battery life or size is also affected.
Input impedance	How much impedance the hybridized detector faces when bonded with the ROIC	Determines the injection efficiency of the detector. It should match with detector impedance for better injection efficiency.
Linearity	Response of the ROIC to incoming flux with respect to varying input current and integration time	Important parameter to identify the illumination level. For successful images requiring high contrast, linearity should be high.
Non-uniformity	Response of different pixels in the ROIC to the same scene	Ideally all pixels should respond equally to the same scene. Layout issues, bias and power supply distribution, etc. can cause non-uniformities. Can be corrected by non-uniformity correction techniques.
Pixel pitch	Size of the pixel	For better spatial resolution and large array format, pixel pitch should be small
Frame rate	Number of scenes displayed in a second	30 Hz is enough for the human eye. For some applications such as missile seeker, high frame rates such as 400 Hz is required.
SNR	Signal-to-noise ratio of the ROIC	An important parameter for the sensitivity of the system. Connected with the noise parameter and charge handling capacity.
Charge handling capacity	Number of charges that the ROIC can store in a pixel	A limiting parameter for high illumination level applications. Pixel should not be saturated by high flux
Gain	Gain of the input pre-amplifier	An adjustable parameter for the analog ROICs for high and low illumination levels. For example, if pixels are saturated, the gain parameter should be decreased.
Dynamic range	Ratio of the maximum detectable signal to the minimum detectable signal	An important parameter for the high contrast applications requiring to see very dark scenes and very bright scenes at the same time

1.4 Thesis Objectives

Most of the available TDI ROICs in the market and literature are analog ROICs and they have some limitations such as charge handling capacity, SNR, pixel pitch, signal integrity and power consumption. There are also a few digital TDI ROIC examples in the literature, most of which are column-parallel ADC based designs. These limited digital TDI ROICs also does not solve all the before-mentioned limitations. The primary objective of this thesis is to answer these limitations and broaden the use of digital ROIC architectures to TDI ROIC implementations. To meet this objective, this thesis proposes a digital readout integrated circuit (DROIC) implementing time delay and integration (TDI) for scanning type LWIR infrared focal plane arrays (IRFPAs). The proposed digital TDI ROIC comes with advantages of improved signal-to-noise ratio (SNR), lower power consumption, smaller pixel pitch compared to analog and digital counterparts. The proposed digital TDI ROIC enables high charge handling capacity together with reduced quantization noise, hence improves SNR significantly. The proposed DROIC converts the photocurrent to digital data in pixel and pixel data is transferred to peripheral circuitry in digital domain resulting in improvement of the signal integrity compared to the analog ROICs, which transfer sensitive analog signals to peripheral circuitry in the analog domain. The proposed DROIC also enables smaller pixel pitch due to the summation of digital pixel outputs at off-pixel counters, which in turn prevents oversampling TDI required for large pixel pitch implementations and saves a considerable amount of chip area. Finally, the proposed DROIC reduces power consumption since it has digital outputs unlike analog ROICs requiring high swing analog outputs.

1.5 Thesis Overview

This thesis is organized into five chapters. After a short summary of infrared imaging systems, ROICs, chapter 2 focuses on TDI implementation and TDI ROIC architectures, namely analog TDI implementations and digital TDI implementations. First, many different analog TDI ROIC examples are given and their drawbacks are discussed. Following the analog ones, digital TDI ROICs and some digital ROICs as potential candidates for TDI implementation in the literature are given and discussed.

The third chapter is dedicated to the implementation of the prototype chip for verification of the proposed architecture. After giving the block level implementation of the whole architecture implementing TDI with an oversampling rate of 2, individual blocks are given in details including both schematic and layout designs.

Chapter 4 gives the simulation and measurement results of the proposed architecture. First, simulation results verifying some individual blocks and whole architecture are given. After that measurement setup is explained and measurement results are given for the verification of the prototype array and proposed idea.

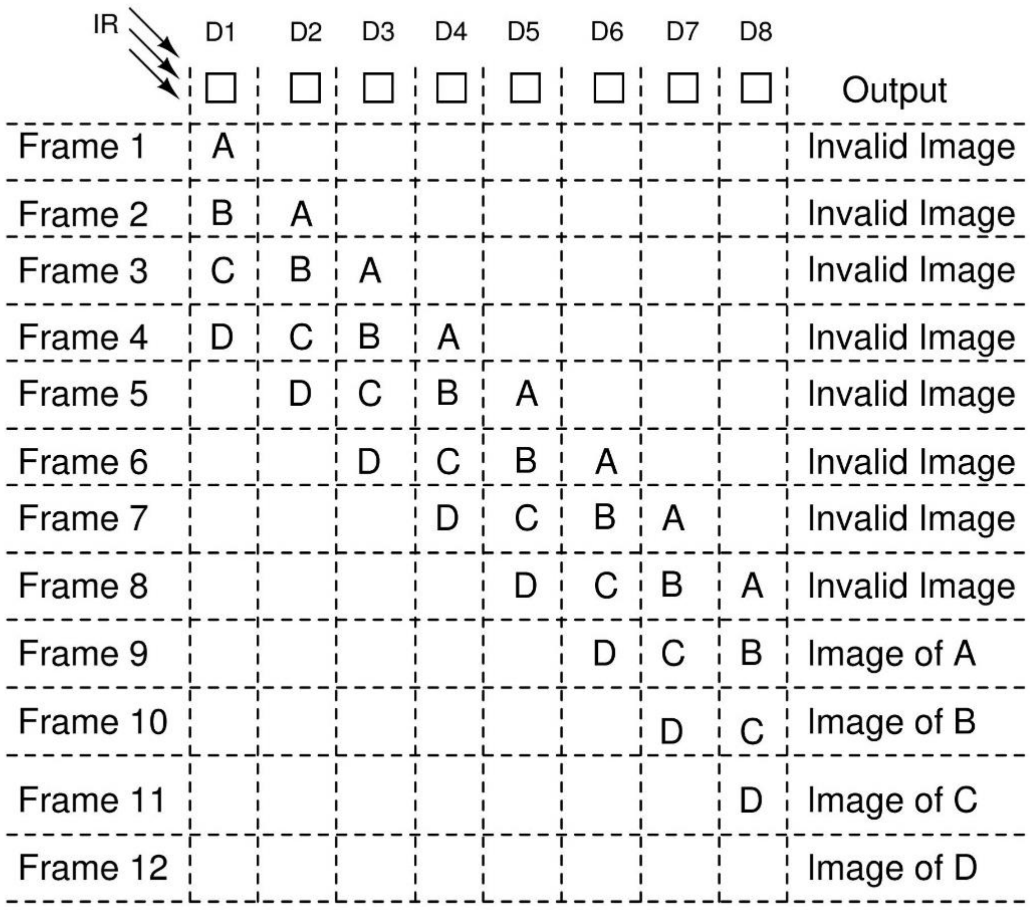
The last chapter summarizes the need for digital TDI ROIC and gives the measured parameters of the prototype array in table format. Comparison of the proposed architecture with some analog and digital TDI ROICs is also given in this chapter. Finally, possible improvements for future work is discussed.

2. REVIEW of TDI ROIC ARCHITECTURES

2.1 TDI Operation

Although third generation 2D IR FPAs are very popular there is still need for scanning array systems. Earth observation from a satellite, products moving on a conveyor belt, reconnaissance from an aircraft naturally include the scanning operation, there is no need for an additional mechanical scanner like in the second generation infrared imaging systems. A satellite scans the earth, an aircraft scans the area that it is flying over and the conveyor belt is already moving to carry products. Some applications also require tracking fast moving objects and operation in low light conditions. Time delay integration (TDI) method that is a technique incorporated in scanning array systems answers these demands. TDI, which is commonly used in scanning array systems, is used to increase the signal-to-noise-ratio (SNR) when imaging fast moving objects [30] and enables low noise imaging under low illumination condition [31]. TDI is implemented by multiple integrations of the same scene on several detector pixels with the help of a scanning array. The scanning optic directs the IR radiation of the same scene to several detectors on the same linear array, thus same image data is stored on different elements. The stored data on different elements are then summed up and constitutes the desired image with an improved SNR. With the use of the TDI, SNR improvement ratio compared to a single pixel operation is the square root of the number of elements in the line [32]. *Figure 9* shows how the TDI operation works on 8 elements. Image of the letter "A" impinges on D1 first and continues on other detectors in the following frames. At frame 8, the letter "A" impinges on D8 and at this point, the image of the letter "A" coming from all 8 detectors are available. Since all data is available after frame 8, at frame 9 all contributions from 8 detector elements are summed up and averaged, which is the output data for letter "A". Similarly, all other letters that impinge on D1 in the following frames are sent to output following the output of letter "A". As it is seen from the figure, outputs in first 8 frames are not valid. Since there is no enough data (contributions of all 8 TDI elements) to sum up in first 8 frames for any of the letters (images), there is no valid data available in first 8 frames. TDI operation is very useful for the applications that there has limited amount of integration time due to the system requirements such as the scanning array systems. For example, if a scanning array system works at 50 Hz frame rate and has

576 rows, it means that in each second all of these 576 lines should output their data 50 times, which means approximately 35 μs maximum integration time. This limited integration time results in a poor SNR value due to less amount of collected electrons. TDI method is a way of extending the integration time which helps to collect more electrons, hence to have improved SNR value. An example for the limited integration time applications is the fast moving scenes. If imaging system moves fast and it is not possible to integrate for a long time like in the case of an earth observation satellite or aircraft flying over an area, TDI becomes one of the best solutions to capture the scene since it has the capability of multiple short integrations on different TDI elements. TDI is also useful for low light level applications that typical single pixel line scan cameras cannot make a useful image [33]. By collecting more photon flux on different TDI elements (detector pixels) a TDI camera can create useful images.



	D1	D2	D3	D4	D5	D6	D7	D8	Output
	□	□	□	□	□	□	□	□	
Frame 1	A								Invalid Image
Frame 2	B	A							Invalid Image
Frame 3	C	B	A						Invalid Image
Frame 4	D	C	B	A					Invalid Image
Frame 5		D	C	B	A				Invalid Image
Frame 6			D	C	B	A			Invalid Image
Frame 7				D	C	B	A		Invalid Image
Frame 8					D	C	B	A	Invalid Image
Frame 9						D	C	B	Image of A
Frame 10							D	C	Image of B
Frame 11								D	Image of C
Frame 12									Image of D

Figure 9: TDI operation with 8 elements

2.2 Analog TDI Implementations

Due to the technological limitations, first generation ROICs designed for scanning array TDI applications were utilizing analog design techniques and there are many different analog TDI ROIC implementations in the literature. Charge coupled devices (CCDs) can be accepted as the first analog TDI implementations. CCDs operate on charge domain meaning that charges are collected by controlling the CCD cells with timing signals, propagated and summed to form the final output. However, CCD circuits require non-standard semiconductor processing [34] and high voltage operation which means high power consumption. The main advantage of CCDs is almost noiseless charge transfer mechanism [35]. CCDs are still used in many commercial products, but with the advance in CMOS process and CMOS image sensor technology, CCDs start losing their market share. It is projected that in 2017, CCD market share will be 15% while CMOS market share will be 85% [36].

Apart from the CCD technology which is not a standard CMOS process, there are many analog CMOS TDI ROIC implementations. Analog TDI ROICs employ a different type of input pre-amplifiers depending on the wavelength and application to collect incoming photocurrent, convert it to voltage domain, store contributions of each TDI element, add contributions of each TDI element and finally multiplex the summed up row data to output to be processed by an ADC on proximity board. One way of implementing this idea is to use multiple storage elements like capacitors to store the data of the same scene on different TDI elements (pixels) until this data is available to all TDI elements. After the data coming from all TDI elements on separate capacitors are available, they are switched and added in a single summing amplifier and sent to the output. Examples of this method can be found in [37] and [38]. This method employs single amplifier and consumes less power compared to the method employing multiple amplifiers. The main drawback of this method is the large chip area due to many storage elements required. *Figure 10* shows the implementation of an analog TDI ROIC. If oversampling TDI is implemented in this configuration, a lot of TDI storage elements are needed, hence analog signal routings from the pixels to the storage capacitors are very long sensitive lines, which are immune to crosstalk or interference.

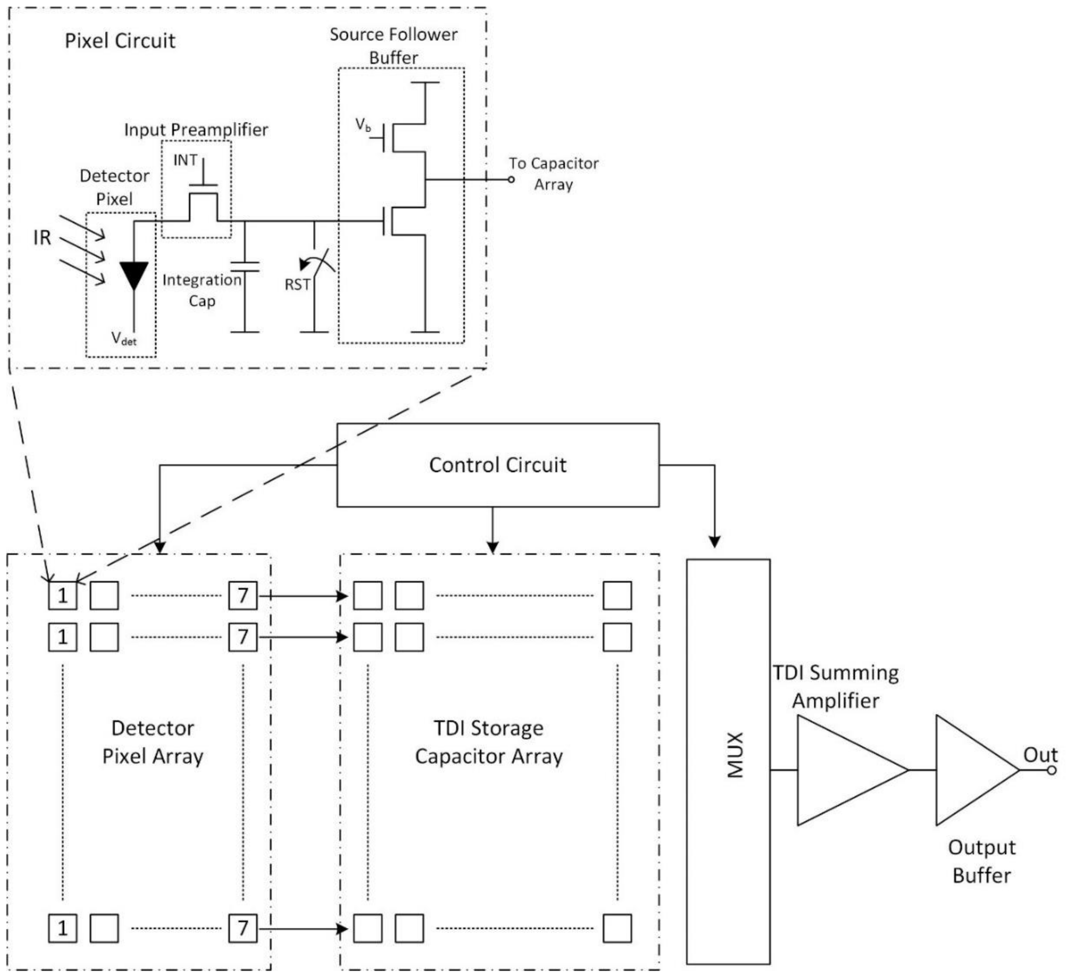


Figure 10: Analog TDI ROIC example

The other way of implementing TDI is to use multiple adders. Each adder is dedicated to a certain scene. When the same scene comes to a TDI element, this data is transferred to the dedicated adder and summed there. This operation is repeated for all TDI elements and when data from all TDI elements are transferred to dedicated summing amplifier to that specific scene and summed there, at this point the output data is available [39] [40] [41]. This method requires less chip area compared to the previous one, but still needs a considerable amount of chip area. Furthermore, offset and gain variations of summing amplifiers together with capacitor variations bring extra noise which is not easy to correct. To solve these difficulties some other methods such as current mode operation and addition is offered. In this method, the contribution of each TDI element is converted to the current domain and amplified in the pixel, and then added in the current domain, which eliminates the use of capacitors in summing amplifiers [42]. However, the current mode operation and the addition of pixels also suffer from mismatches and process variations. Additionally, this method suffers from low dynamic range. It is not possible to handle very low and high flux scenarios at the same time with this method.

Another example of the analog TDI implementations is the adjacent pixel signal transfer architecture similar to the CCD operation. In this method, each TDI element contribution is transferred to the adjacent pixel with the help of an off-pixel amplifier [43] or an in-pixel amplifier [34]. This method also suffers from the low dynamic range, since it is suitable mostly for CMOS image sensors with very low input current. In this method, pixel capacitor can easily be saturated with high input current levels. A different version of the adjacent pixel signal transfer method is described in [44]. In this method, an analog sampling stage, which includes an amplifier and two capacitors, is put in the pixel. The analog sampling stage integrates the signal of the previous analog sampling stage with the sampled photo signal of the corresponding pixel and subtracts the reset level, hence in-pixel correlated double sampling (CDS) is also included in this architecture.

Another analog TDI method proposed to improve the dynamic range of analog TDI ROICs is the self-adaptive scanning architecture [45]. In this architecture, there is an additional calibration pixel detecting the incoming flux and deciding which setting will be used for the TDI pixels next to the calibration pixel. Depending on the flux level, capacitance values of the pre-amplifiers are adjusted, hence dynamic range of the TDI ROIC is improved. All of these analog TDI architectures suffer more or less from the sensitive analog signal routing problems. There are also some methods to minimize these signal routing effects. Since all pixels in a column share a long output bus and there are many digital control signals and power lines passing nearby these sensitive lines, it is very important to carefully route these signals or make them immune to any interference. An improved version of CDS pixel and a way of routing sensitive column bus signal is proposed in [46]. In this method, conventional two capacitor CDS implementation is improved with a single capacitor CDS implementation. Additionally, sensitive column bus signal is protected from the parasitic effects of digital signals passing nearby by adding the inverse of the digital signals to the other side of the column bus.

As it seen from the above examples analog TDI ROIC implementations require careful signal routing, are limited in dynamic range and generally consumes high power due to high swing amplifiers required to improve the dynamic range. They are also limited in charge handling capacity and occupy large chip area.

2.3 Digital TDI Implementations

Development in digital readout technology also has an impact on TDI readout techniques and digital TDI ROICs are started to be implemented. Although digital ROICs have been gaining ground in recent years, there are only a few digital TDI ROIC implementations in the literature [47] [48] [49]. Lepage et al. gives an overview of TDI architectures for CMOS image sensors in his paper and describes some techniques for the TDI readout architectures [47]. The paper applies column digital readout technique to TDI applications and proposes a technique to implement the TDI functionality for CMOS image sensors.

Figure 11 shows the proposed digital TDI architecture for CMOS image sensors. Each column of pixels is connected to a column ADC with enough amount of memory locations. The addition of pixel contributions is done in ADC up counter. *Figure 12* shows the connection sequence of pixels and memory locations/adders. Adders in *Figure 12* can be thought as memory locations, because at the end of each frame the counter value, that is the addition of previously connected pixels, is stored in a memory location. Each vertical line with stars indicates the beginning of a new frame. In each frame, each pixel is connected to a different adder and at the end of 5 frames, all 5 pixels are connected to the corresponding adders, which means output for that particular scene is ready to be multiplexed. *Figure 11* also shows column ADC structure together with the memory locations used for the digital TDI implementation. A ramp signal is compared with the pixel output and counter counts the clock cycles until the comparator is triggered. The comparator is triggered when the ramp signal and the pixel output are at the same voltage level. After the frame ends, the counter value is stored in a memory location dedicated to the particular scene. When another pixel contribution to that particular scene available, the value stored in the memory location is

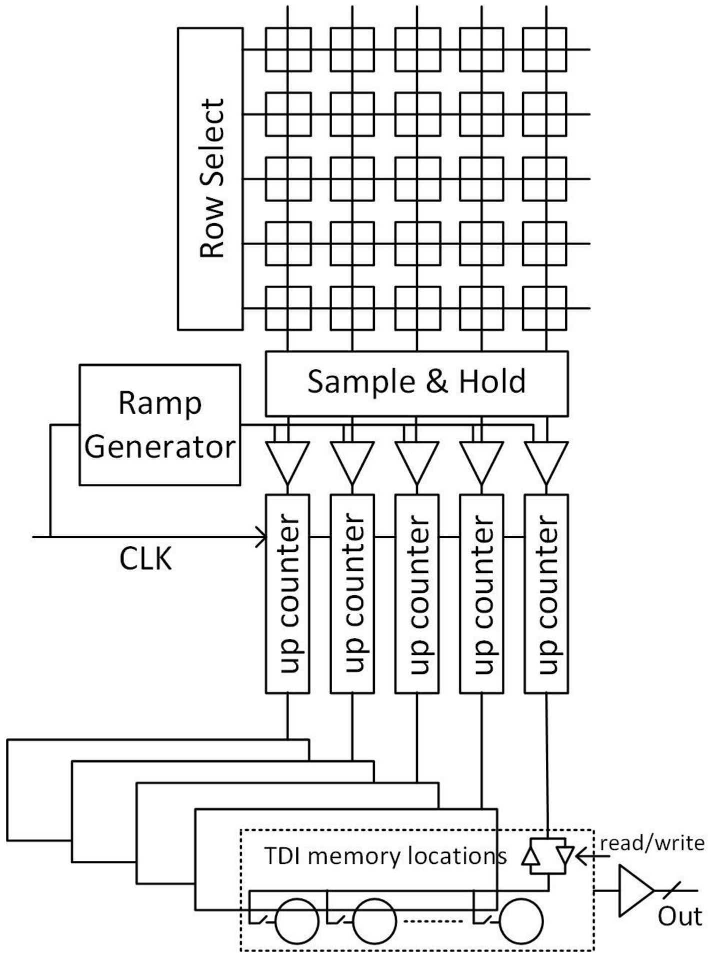
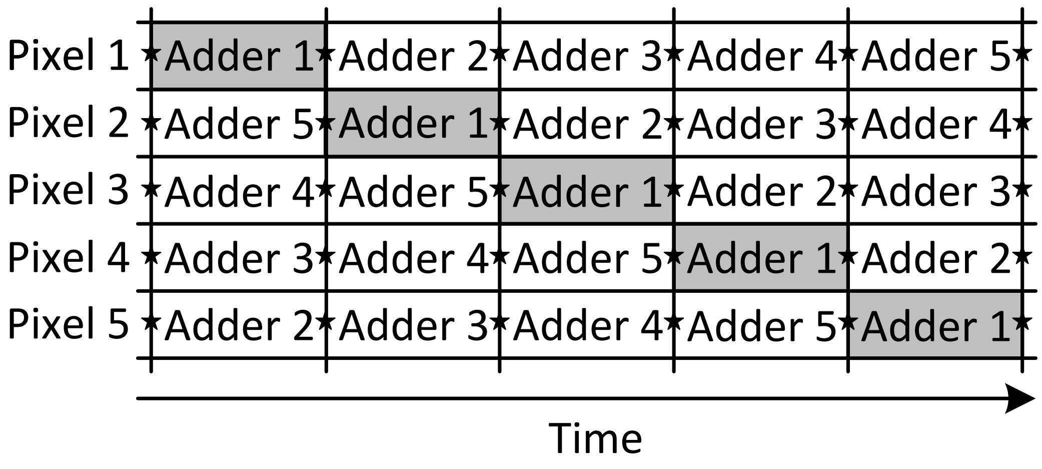


Figure 11: Digital TDI ROIC example *[48]*



Pixel 1	Adder 1	Adder 2	Adder 3	Adder 4	Adder 5
Pixel 2	Adder 5	Adder 1	Adder 2	Adder 3	Adder 4
Pixel 3	Adder 4	Adder 5	Adder 1	Adder 2	Adder 3
Pixel 4	Adder 3	Adder 4	Adder 5	Adder 1	Adder 2
Pixel 5	Adder 2	Adder 3	Adder 4	Adder 5	Adder 1

Figure 12: Timing of digital TDI implementation of *[47]*

called back to the counter. The counter continues to count from that value. In this way, there is no need for a real adder. The addition is done by calling the previously stored value and continuing the count from this stored value. There is a switch controlling read/write operation from the column parallel ADC counter to the memory location. The detailed operation of the column-parallel digital TDI architecture here is described in US patent [48] and [50].

The counter used in the column-parallel ADC is the unidirectional counter. In the unidirectional counter architecture, the ramp signal starts falling above the predetermined reset value V_{reset}. There is a control circuit that controls the count operation. If the ramp signal is above the reset value V_{reset}, the control circuit does not let the counter count. After the ramp signal reaches the reset value, the counter starts counting. There is also a bidirectional counter version of the column parallel ADC architecture [49]. Correlated double sampling (CDS), which is used to eliminate reset noise, is done in a different way in the bidirectional counter version of the column parallel ADC architecture. The counter counts downward while capturing the reset signal and counts upward while capturing the actual signal, in this way the noise signal is subtracted from the total signal. *Figure 13* shows the operation of the unidirectional counter and the bidirectional counter architectures for the TDI operation.

Another version of the column parallel ADC architecture for TDI implementation is to use two sets of sampling capacitors and an optimized timing together with a cyclic ADC [51]. A 128-stage CMOS TDI image sensor is implemented with an on-chip digital accumulator. The TDI adding operation is done in the digital accumulator. Another property of this architecture is the use of CDS architecture in order to reduce the noise, which is common practice for CMOS image sensors. The digital accumulator that is connected to each column of pixels, includes a 17-bit adder and 129 memory cells in order store the 128-stage pixel contributions.

The last column parallel ADC architecture for digital TDI implementation is described in the US patent [52]. The only different thing in this patent compared to the previous US patents, is the use of a column adder instead of the counter. Every time a pixel contribution is required to be added to the previous pixel contributions of the same scene, previously stored value is called back from the memory and added with the newest pixel contribution and stored back to the memory. Since the adder will be occupying a larger area than a unidirectional counter, this architecture is not area efficient as the previous method. It is also worth to mention that column-parallel ADCs used in almost all of these architectures convert the pixel output to the digital domain in terms of time, which means they incorporate clock signals.

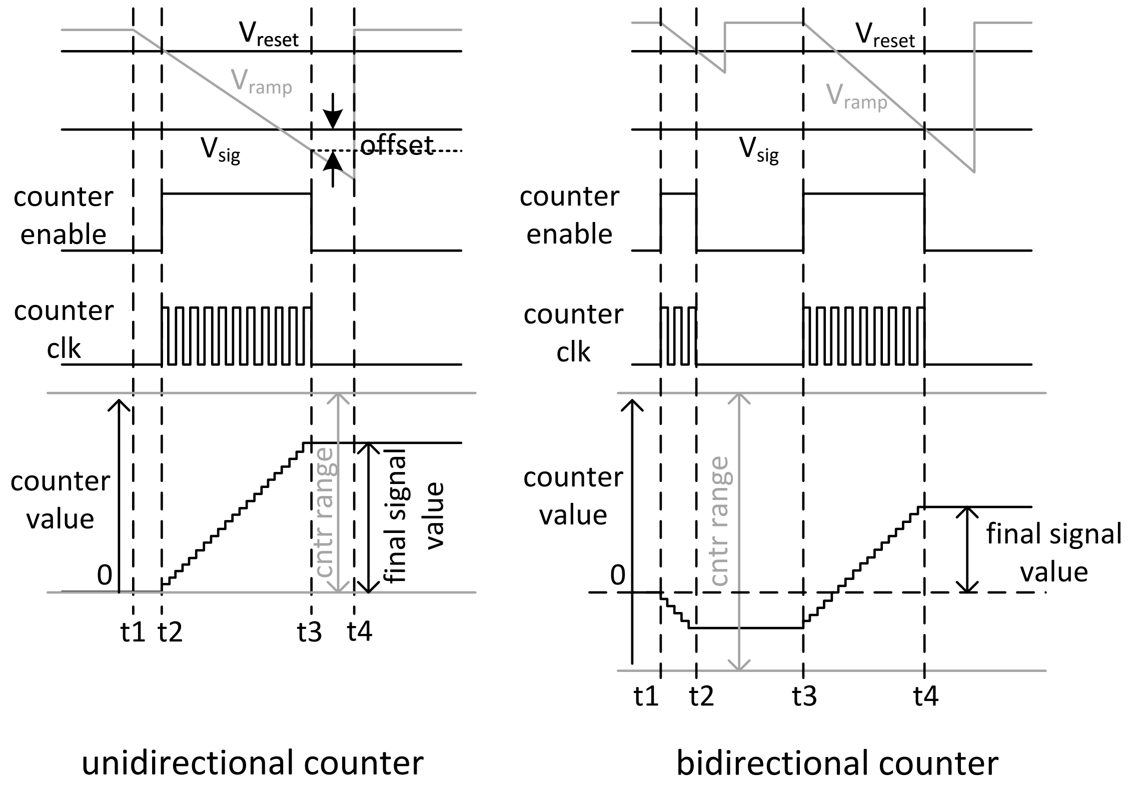


Figure 13: Digital TDI implementation with unidirectional and bidirectional counter *[48]*

There is also pixel level digital TDI implementation using the pulse-frequency-modulation technique together with the orthogonal transfer that is shown in *Figure 14* [53] [54]. A simple PFM circuit is shown with the direct injection input preamplifier and the orthogonal transfer structure. To minimize the quantization noise, a 1fF very small capacitor is used as an integration capacitor, which means increased number of comparator triggers, hence more power consumption. Although a very small integration capacitor is used, quantization noise is still high due to the charge packet size of approximately 3000 electrons assuming 0.5V reference voltage. In this configuration, the capacitor voltage is compared with a reference signal V_{ref}, and if the capacitor voltage is same as V_{ref}, the comparator is triggered and a counter stores the number of counts in each frame. TDI operation is realized by using the orthogonal transfer method. Orthogonal transfer structures incorporate registers and a multiplexer, and transfers the digital data stored in the counters to the left, right, up or down registers. Although advanced node CMOS process technology is used in this digital pixel sensor architecture, it is still not possible to have a small pixel pitch. Because, the comparator and the reset circuit together with the counter and the orthogonal transfer register occupy a large pixel area. Details of the orthogonal transfer and different PFM implementations are explained is US patent [54].

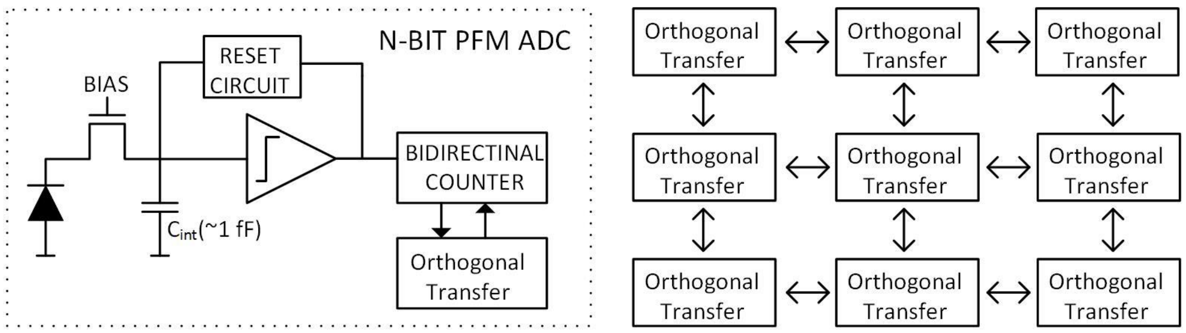


Figure 14: Pulse Frequency Modulation (PFM) Architecture for digital pixel readouts *[53]*

Different from previously described DROIC implementations, there are also some methods, combining PFM technique with some other techniques to reduce quantization noise. One of them is called time-frequency fusion digital pixel sensor, combining pulse-frequency-modulation technique and time-to-digital conversion technique [55]. Most-significant-bits of the final digital output are created by the PFM technique, whereas the least-significant-bits of the final digital output are created by the time-to-digital conversion technique. The other technique to reduce the quantization is the extending counting technique, which uses PFM technique for MSB bits while using extending counting method for LSB bits [56]. However, these DROIC techniques are not implemented for TDI applications.

3. IMPLEMENTATION

This chapter gives the implementation of the proposed digital TDI architecture both in system level and block level. First, system level ROIC architecture is introduced with the implementation of TDI with an oversampling rate of 2. After that, the pixel design, the ramp generator, counters including the multiplexers and the driving circuits, the control circuit, the serializer, test circuits including the current source used to mimic the detector input current are explained in detail with their schematics and layouts. *Figure 15* shows the category of the proposed digital TDI ROIC among the IRFPAs. The proposed digital TDI ROIC is implemented for multiple elements (8 pixels) scanning type LWIR IRFPA.

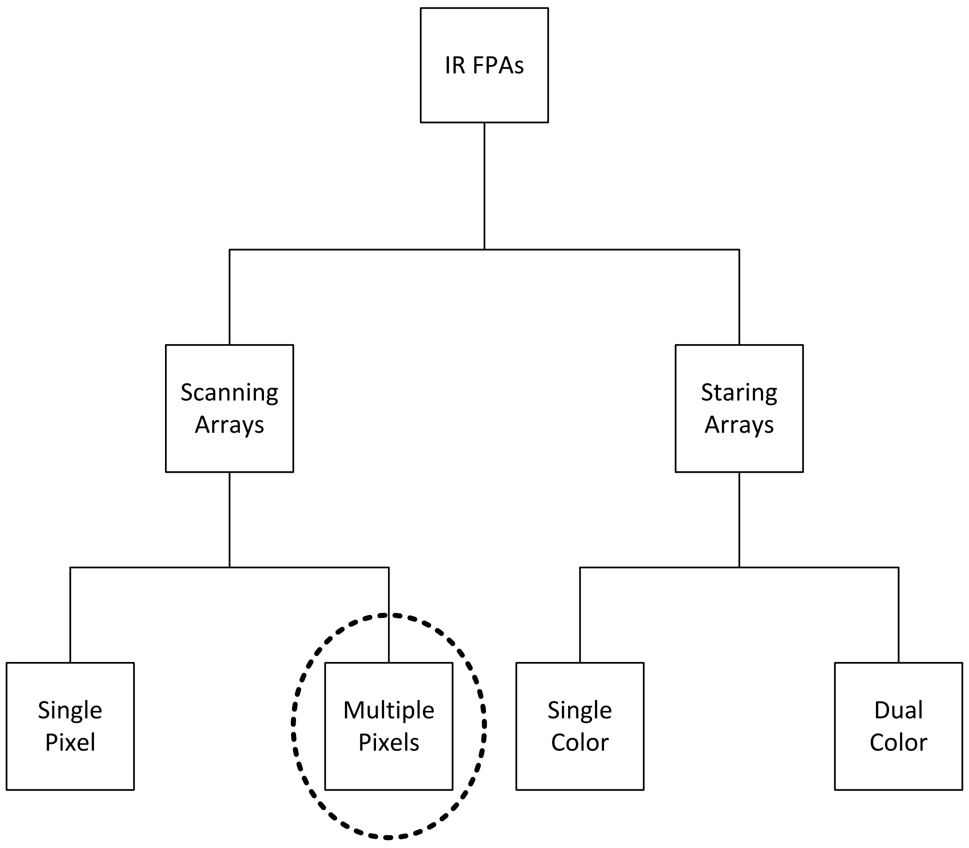


Figure 15: Block diagram showing the category of the proposed digital TDI ROIC

3.1 The Proposed Digital TDI ROIC Architecture

The proposed digital TDI ROIC is based on the conventional PFM architecture that is shown in *Figure 14*, but is improved in terms of the quantization noise for better performance in the low illumination conditions. The proposed method combines the use of the simple PFM architecture for the coarse quantization and the single slope ADC architecture for the fine quantization. Unlike the digital pixel sensors, digital storage elements are put outside the pixel area that is special to the digital TDI implementation. Putting the storage counters outside the pixel area gives the opportunity to have a smaller pixel pitch and to use less oversampling rate. In that sense, the proposed digital TDI architecture is a hybrid solution that puts the advantages of the column parallel ADC architectures and the digital pixel architectures. Column-parallel digital TDI architectures convert the analog pixel output to the digital domain at the end of the column, not in the pixel, and stores the converted data in the registers nearby the pixel array in the same row. On the other hand, digital pixel readouts store all of the information in the pixel, hence need more pixel area to have a reasonable number of output bits for high dynamic range, which means larger pixel area. The proposed technique converts the analog pixel output to digital domain in the pixel, hence it is possible to route the digital signal to any distance due to immunity of digital signals to any interference. The proposed technique also does not route the clock signal to the pixel area for the time-to-digital conversion, that is required to generate LSBs. Since the clock signal is not routed to the pixel area, this will improve the noise performance of the pixel, which is a major problem in digital pixel readouts. Instead of using a very small integration capacitor as in the conventional PFM architectures, a relatively large integration capacitor of 40fF is chosen in order to have less number of comparator triggers, hence less power consumption. Despite using a large integration capacitor, quantization noise is kept low with the use of a single slope ramp ADC. *Figure 16* shows the proposed digital TDI ROIC architecture and *Figure 17* shows the timing diagram of the proposed architecture.

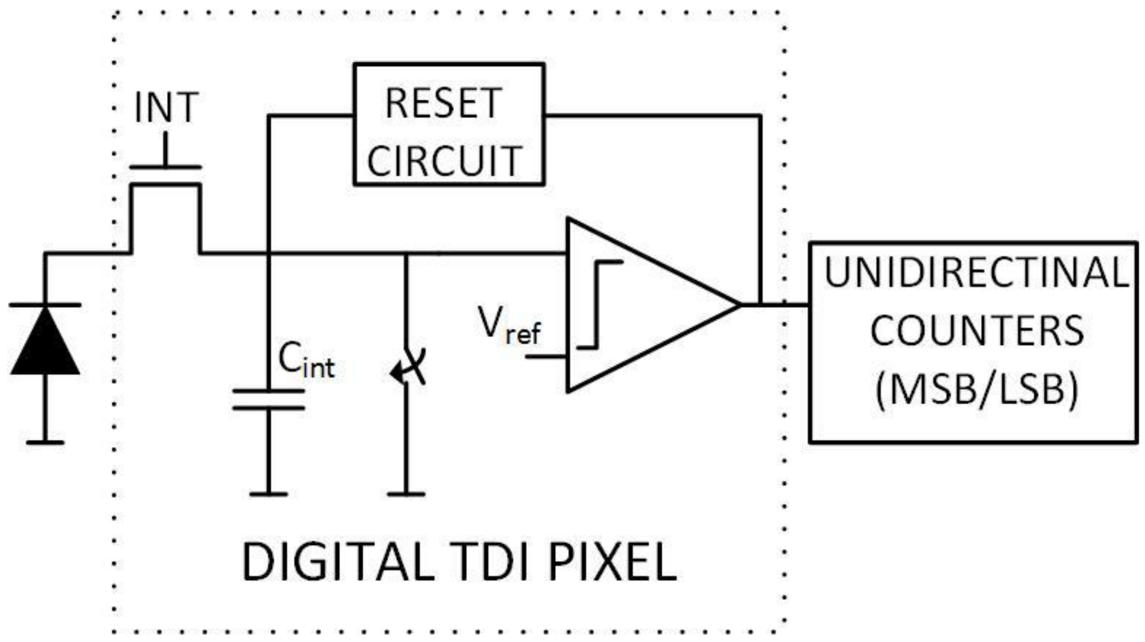


Figure 16: The proposed digital TDI pixel

The proposed digital TDI architecture has two phases, namely the integration phase and the residual phase. A ramp generator generates a ramp signal starting from the reset level reaching the reference value during the residual phase and the reference value is fixed during the integration phase. Integration capacitor voltage is compared with the fixed reference voltage during the integration phase. When two signal values become equal, comparator switches and the integration capacitor is auto-reset by the reset circuit. Each time the comparator is switched, the corresponding MSB counter is also triggered. The number of comparator triggers determine the MSB counter output. At the end of the frame, there will be residual charges remaining on the integration capacitor due to quantized charge packets. These residual charges on the integration capacitor are measured by the single slope ramp ADC in the residual phase. Ramp generator output is compared with the residual voltage on the capacitor in that phase. If the integration capacitor voltage is greater than the ramp generator output voltage, the LSB counter keeps counting the number of clock cycles. When the ramp generator output is equalized with the residual voltage, the comparator is triggered and the LSB counter stops counting the clock cycles. The ramp generator sweeps the whole dynamic range in 256 clock cycles. For example, if the residual charge on the integration capacitor is half of the full charge packet, that means 128 clock cycles will be counted by the LSB counter.

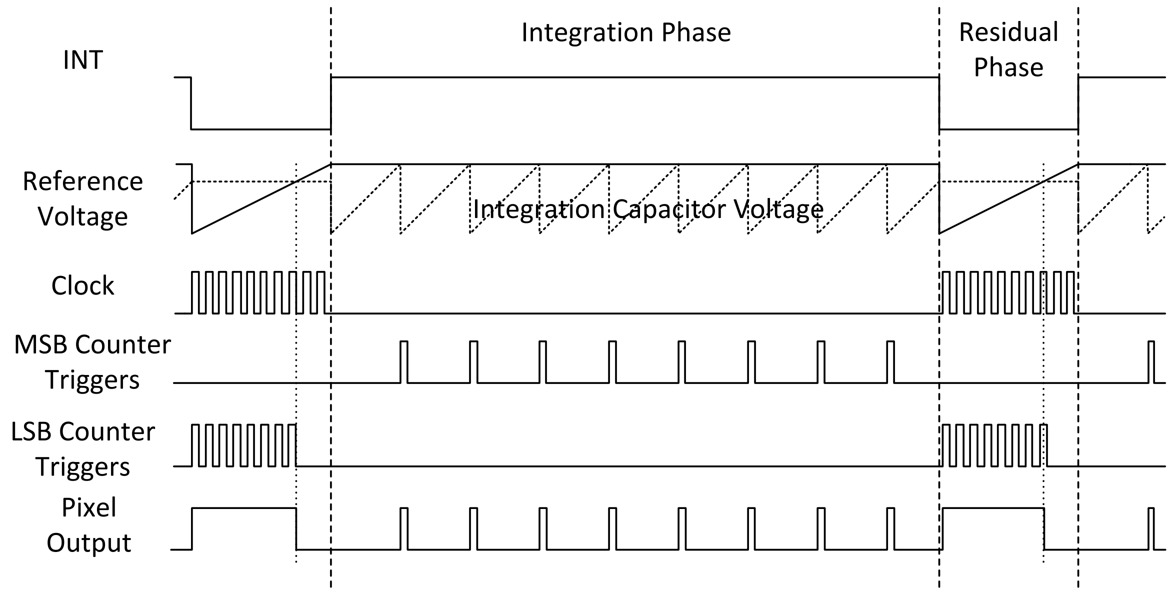
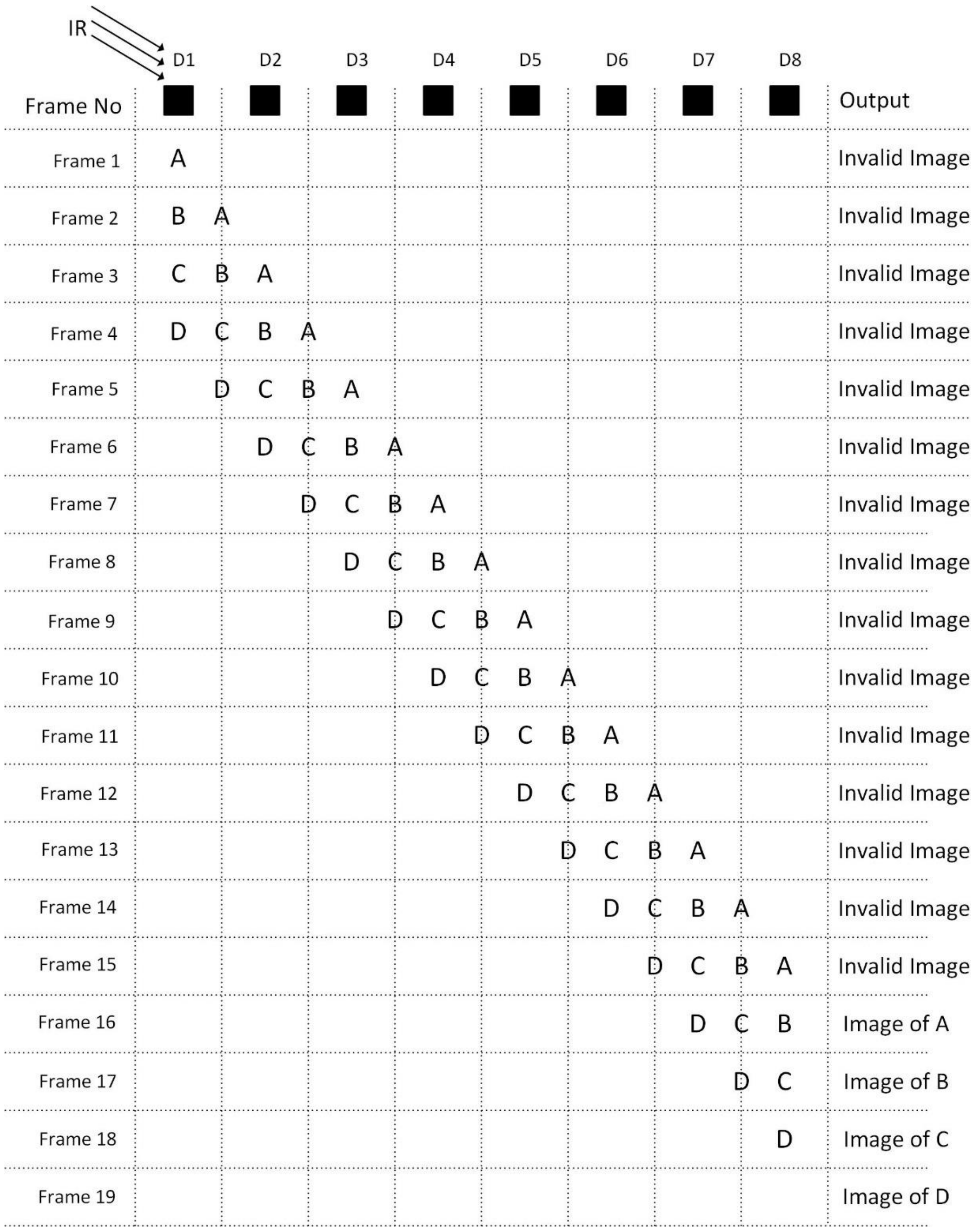


Figure 17: Timing diagram of the proposed digital TDI architecture

Apart from the pixel implementation of the digital TDI architecture, implementation of the TDI functionality in architectural level is also very important. Each output should contain all pixel contributions in order to have an improved SNR. Implementing the TDI functionality is easy if there is no oversampling. Because each counter value can be shifted by one and the next pixel contribution can be added to this pre-set value as in [53]. However, it is becoming a complex issue when oversampling comes into the picture. The oversampling rate stands for the number of steps that is required for a particular scene to move from one pixel to the adjacent one. For example, if the oversampling rate is 2, that means a particular scene moves from one pixel to the next in 2 steps. *Figure 18* shows how scenes are propagating and the final output is generated for an oversampling rate of 2, which is the same rate for the proposed architecture. It takes 16 frames for a particular scene to pass over all 8 pixels and to be transferred to the output, which means 16 different counters are needed to store the information.



Frame No	D1		D2		D3		D4		D5		D6		D7		D8	Output
Frame 1	A															Invalid Image
Frame 2	B	A														Invalid Image
Frame 3	C	B	A													Invalid Image
Frame 4	D	C	B	A												Invalid Image
Frame 5		D	C	B	A											Invalid Image
Frame 6			D	C	B	A										Invalid Image
Frame 7				D	C	B	A									Invalid Image
Frame 8					D	C	B	A								Invalid Image
Frame 9						D	C	B	A							Invalid Image
Frame 10							D	C	B	A						Invalid Image
Frame 11								D	C	B	A					Invalid Image
Frame 12									D	C	B	A				Invalid Image
Frame 13										D	C	B	A			Invalid Image
Frame 14											D	C	B	A		Invalid Image
Frame 15												D	C	B	A	Invalid Image
Frame 16													D	C	B	Image of A
Frame 17														D	C	Image of B
Frame 18															D	Image of C
Frame 19																Image of D

Figure 18: Implementation of TDI functionality with oversampling rate of 2

Table 6: Write/read sequence of counters for TDI operation with oversampling rate of 2

Frame No	**C 1**	**C 2**	**C 3**	**C 4**	**C 5**	**C 6**	**C 7**	**C 8**	**C 9**	**C 10**	**C 11**	**C 12**	**C 13**	**C 14**	**C 15**	**C 16**
1	P1															
2		P1														
3	P2		P1													
4		P2		P1												
5	P3		P2		P1											
6		P3		P2		P1										
7	P4		P3		P2		P1									
8		P4		P3		P2		P1								
9	P5		P4		P3		P2		P1							
10		P5		P4		P3		P2		P1						
11	P6		P5		P4		P3		P2		P1					
12		P6		P5		P4		P3		P2		P1				
13	P7		P6		P5		P4		P3		P2		P1			
14		P7		P6		P5		P4		P3		P2		P1		
15	P8		P7		P6		P5		P4		P3		P2		P1	
16	**R**	P8		P7		P6		P5		P4		P3		P2		P1
17	P1	**R**	P8		P7		P6		P5		P4		P3		P2	
18		P1	**R**	P8		P7		P6		P5		P4		P3		P2
19	P2		P1	**R**	P8		P7		P6		P5		P4		P3	
20		P2		P1	**R**	P8		P7		P6		P5		P4		P3
21	P3		P2		P1	**R**	P8		P7		P6		P5		P4	
22		P3		P2		P1	**R**	P8		P7		P6		P5		P4
23	P4		P3		P2		P1	**R**	P8		P7		P6		P5	
24		P4		P3		P2		P1	**R**	P8		P7		P6		P5
25	P5		P4		P3		P2		P1	**R**	P8		P7		P6	
26		P5		P4		P3		P2		P1	**R**	P8		P7		P6
27	P6		P5		P4		P3		P2		P1	**R**	P8		P7	

Table 6 shows the write/read sequence of the MSB and LSB counters for the TDI operation with an oversampling rate of 2 for the first 27 frames. P1 to P8 represents the TDI pixels. In each frame, each pixel is connected to a different counter. Two consecutive pixels write to a counter with one frame interval due to the oversampling rate of 2. After all of the pixels write to a counter, read operation is performed for this counter. “R” represents the read operation of that counter in *Table 6*. Although all pixels write to a counter in 15 frames, one more counter is required. The reason of the use of this extra counter is the timing conflict of the read / write operations. Since a read and a write operation cannot be performed at the same frame, this extra counter is used.

Figure 19 shows the TDI architecture with an oversampling rate of 2 for a 90x8 prototype array in agreement with *Table 6*. The proposed digital TDI ROIC architecture is composed of 5 main building blocks, which are pixels, counters, the digital control block, the ramp generator and the serializer. Apart from the 8 pixels, there are also bypass/TDI direction/pixel deselection switches, MSB / LSB counters, their read/write control switches and multiplexers in a row. In bypass mode of operation, each one of the 8 pixels in a row can be selected individually and connected to the corresponding counters, hence dead pixels can be detected individually. The TDI direction switches control the arrangement of pixel connections to corresponding counters. In the positive TDI direction, the first pixel is connected to the first counter like in *Table 6*. However, in the negative TDI direction, the 8th pixel becomes the first pixel, which means pixels are connected in reverse order to the counters. This feature proves useful for the scanning array systems, which enables the system to operate in both scan directions. Pixel deselection switches are used to select which pixels are connected to the counters. Defective pixels detected in bypass mode are switched-off and not connected to the counters. Both the MSB and LSB counters are ripple carry counters implemented with D flip-flops. Each counter has a multiplexer controlled by the digital control circuit to select which pixel is connected to this counter in a particular frame. Each MSB and LSB counter have 11-bit registers that are required to add 8-bit contributions of 8 different pixels in a row. There are 16 MSB and LSB counters required to store the data related to 16 different scenes as presented. MSB counters and LSB counters have separate 11-bit output buses for each row. When a row is multiplexed by the control circuit to the output bus, 22 bits are connected to the serializer, and the serializer shifts these 22 bits one by one as a serial output. Apart from the blocks mentioned above, there is also a ramp generator block that is realized by an integrator op-amp, which supplies the reference voltage to all pixels during the integration and residual phases. The digital control circuit supplies the digital signals for all blocks including pixels, bypass/TDI direction/pixel deselection switches, counters, multiplexers, serializer, ramp generator. The digital control circuit can be programmed through either a serial or a parallel interface. Parallel programming capability is used to operate the ROIC in basic operation mode. It includes all bypass and TDI modes of operation with a certain number of pre-defined integration times. If an integration time different than the pre-defined ones required, then it is possible to program the ROIC through the serial interface. Pixel deselection feature is also accessible only through the serial interface.

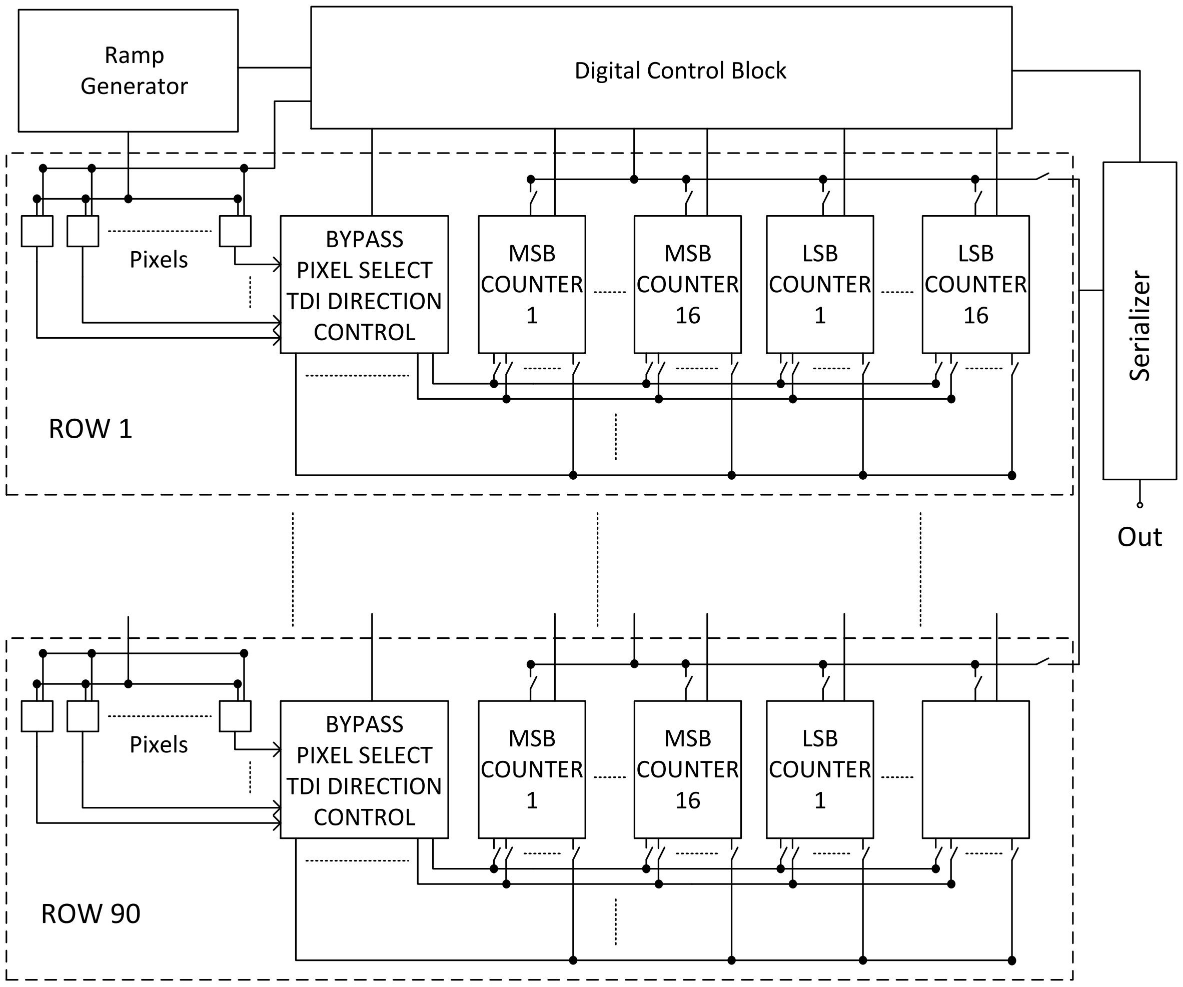


Figure 19: Digital TDI ROIC architecture with an oversampling rate of 2 for a 90x8 array

3.2 Pixel Design

For the pixel design the charge handling capacity, the integration capacitor value, the quantization noise, and the power consumption are to be decided first of all. Although inverter based comparator is used in many PFM architectures due to its small area, less power consumption, and simple design, a self-biased differential amplifier based comparator topology [57] is chosen in order to meet adjustable reference voltage and power consumption requirements. This comparator topology also does not require any external biases, which simplifies the pixel design and routings, is another factor why this topology is chosen. Since the digital TDI ROIC is designed to be operated in cryogenic temperatures (77K) and threshold voltage (V_{th}) increases significantly at cryogenic temperatures, which limits the comparator input swing, 2.5V supply voltage is preferred for the pixel design instead of 1.8V.

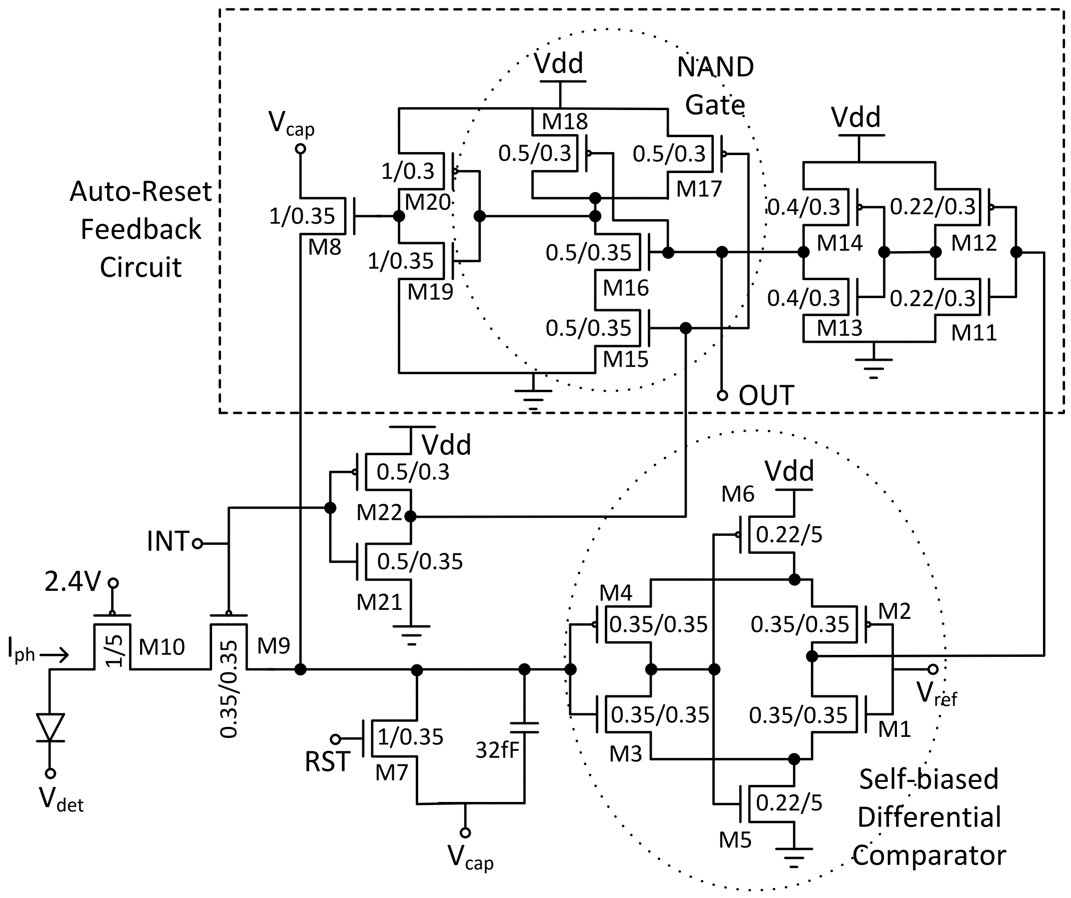


Figure 20: Pixel schematic, transistor sizes in μm

Figure 20 shows the schematic of the proposed pixel design. Input photocurrent is integrated on the integration capacitor when the INT switch is activated. The PMOS transistor connected to the photodetector is the direct injection (DI) pre-amplifier, which is a commonly used input circuit for LWIR applications, which are exposed to high flux IR radiation. The self-biased differential amplifier compares the integration capacitor voltage with the reference voltage. When V_{ref} is same as the integration capacitor voltage, the comparator is triggered and the auto-reset feedback circuit resets the integration capacitor. This process continues until the end of the integration time. There is a NAND gate in the auto-reset feedback circuit, which is used to differentiate the integration phase and the residual phase. When the residual phase starts, this NAND gate prevents the integration capacitor to be reset and at the same time supplies the control signal for the LSB counter. *Figure 17* shows how the pixel output is synchronized with the comparator output in the residual phase. The width of the pixel output pulse in the residual phase is proportional to the residual charge on the integration capacitor. Two back-to-back connected inverters shape the comparator output as a buffer, the output of which is connected to one of the NAND inputs. The output of the NAND gate is connected to an inverter, which is used to activate the auto-reset transistor. Finally, an inverter generates the inverted version of the INT signal, which is connected to the other input of the NAND gate to control the LSB counter write operation.

The first parameter for the pixel to be decided is the integration capacitor size, which is decided as 32fF in order to meet the charge handling capacity, the quantization noise and the pixel pitch specifications. After deciding the integration capacitor size, the charge packet in a single count can be found as:

$$Q = \frac{\mathrm{C_{int}}\times(\mathrm{V_{ref}}-\mathrm{V_{cap}})}{\mathrm{q}} = \frac{40\mathrm{fF}\times 0.7\mathrm{V}}{1{,}602\times 10^{-19}} \cong 175.000\ \mathrm{e}^{-} \qquad (3.1)$$

The integration capacitor in equation (3.1) is taken as 40fF due to parasitic capacitors coming from 5 MOS transistors and routings connected to the integration capacitor node in addition to the 32fF MIM capacitor. After calculating the charge packet for a single count, the number of bits for the MSB counter can be decided considering the charge handling capacity. Conventional TDI ROICs have a charge handling capacity between 10-20 Me-. Thus, 8 bits of MSB counter will be more than enough for a typical scanning array ROIC. In this case, the total charge handling capacity can be calculated as:

$$Q_{total}=2^8 \times 175.000e^- = 44{,}8\ Me^- \tag{3.2}$$

Another important point of the proposed architecture is the number of LSB bits, which will determine the quantization noise. In order to minimize the quantization noise and keep the residual phase as small as possible, the number of LSB bits is chosen as 8 bits. In this case, the quantization noise will be:

$$QN=\frac{C_{int\times}(V_{ref}-V_{cap})}{q\times 256\times\sqrt{12}} \cong 198e^- \tag{3.3}$$

Similarly, the kTC noise per MSB count can be calculated as:

$$n_{kTC}(77K) = \frac{\sqrt{kTC}}{q} = \frac{\sqrt{1{,}38\times 10^{-23}\times 77\times 40\times 10^{-15}}}{1{,}602\times 10^{-19}} \cong 41e^- \tag{3.4}$$

For the given integration capacitor, the number of MSB counts for a given photocurrent can be calculated. The MSB counter value is given as:

$$N_{MSB-counts}(i_{ph}) = {i_{ph} \times t_{int}} \Big/ {C_{int} \times (V_{ref} - V_{cap})} \tag{3.5}$$

where i_{ph} is the photocurrent, t_{int} is the integration time, C_{int} is the integration capacitor, V_{ref} is the comparator's reference voltage coming from the ramp generator and V_{cap} is the capacitor reset voltage. A detailed noise analysis of the PFM pixel can be found in [53]. The proposed architecture is different in two ways from the PFM architecture in [53]. The first difference is the use of the self-biased differential amplifier based comparator instead of the inverter based comparator. This change in comparator design does not affect the noise calculation of the comparator. Dependence of comparator noise to transconductance of the comparator is formulated in the same way in both comparator designs. The second difference is the ramp generator that is supplying the reference voltage to the comparator's other input. The noise of the ramp generator is added to the total noise equation in square roots as an independent noise source. Thus, the total noise in the proposed architecture can be calculated in electrons as:

$$\begin{aligned} n_{e-T-N} = \Bigg\{ \Bigg[\Bigg(& \frac{C_{int}(V_{ref} - V_{cap})}{q} + \frac{kTC_{int}}{q^2} + \frac{e^2_{pa-white} t_{int}}{2q^2 {R_d}^2 N_{counts}} + \frac{8kT{C_{int}}^2 N_{MSB-counts}}{3q^2 g_m t_{int}} \\ & + \overline{V^2_{ramp-gen}} \Bigg) N_{MSB-counts} \Bigg] \\ + \Bigg[i_T^2 \left(\alpha^2_{1/f} + \alpha^2_{comp@1Hz} \right) + i_R^2 \alpha_R^2 + \frac{e^2_{DI_n@1Hz}}{{R_d}^2} \Bigg] & \left(\frac{t^2}{q^2} \right) ln \left(\frac{t_{samp}}{t_{int}} \right) + \frac{{C_{int}}^2 (V_{ref} - V_{cap})^2}{12q^2 256^2} \Bigg\}^{0.5} \end{aligned} \tag{3.6}$$

where q is the electron charge, (V_{ref}-V_{cap}) is the integration capacitor voltage swing, $e_{pa-white}$ is the white noise component of the preamplifier noise, R_d is the dynamic impedance of the detector, τ is the average time per count, g_m is the transconductance of the comparator, k is Boltzmann's constant, T is temperature, i_T is the detector current, i_R is the ramp generator current, and $\alpha_{1/f}$, α_R, α_{comp}, e_{DIn} correspond to the detector, the ramp generator, the comparator and the injection pre-amplifier 1/f noise contribution factors, and t_{samp} is the measurement time. The first bracket corresponds to the white noise components; the shot noise, reset (kTC) noise, preamplifier noise, comparator noise and ramp generator noise. The number of noise electrons in a single count is multiplied by $N_{MSB\text{-}counts}$. The second bracket corresponds to the 1/f noise components from the detector, the comparator, the ramp generator and the injection preamplifier. Finally, the last term corresponds to the quantization noise.

Knowing that the total signal in electrons is:

$$S = \frac{C_{int}(V_{ref} - V_{cap})N_{MSB-counts}}{q} \tag{3.7}$$

The SNR of the pixel can be calculated by dividing the Equation (3.7) to Equation 3.6.

The proposed digital TDI architecture does not include any counters and registers in the pixel unlike the digital pixel sensors. Including the counters and the memory in the pixel requires at least 25 µm pixel pitch, which contradicts with the SWaP trend. Since the proposed digital TDI ROIC is designed for a scanning array system, there is no need to put everything in the pixel like in the staring array systems. It is possible to put the digital circuits nearby the pixel array, which will replace the analog storage elements in analog TDI ROICs. Putting the digital circuits nearby the pixel array enables smaller pixel pitches and improves the spatial resolution. Most of the conventional analog TDI ROICs employ oversampling to improve the spatial resolution of the system, which exacerbates the system complexity. The proposed digital TDI ROIC proves that it is possible to have smaller pixel pitches and reduced oversampling rates or even no oversampling, which enables less complex, less power hungry and less noisy TDI imaging systems. *Figure 21* shows the pixel layout of the proposed ROIC. Pixel dimensions are 15 µm X 15 µm. The reason behind this specific size was the popularity of 15 µm LWIR detectors at the time this work was started. As it can be figured out from the pixel layout, it is possible to have a pixel pitch of 10 µm with a smaller integration capacitor. This smaller pixel pitch of 10µm will also eliminate the use of the oversampling, which will also halve the number of the MSB and LSB counters resulting in a less chip area.

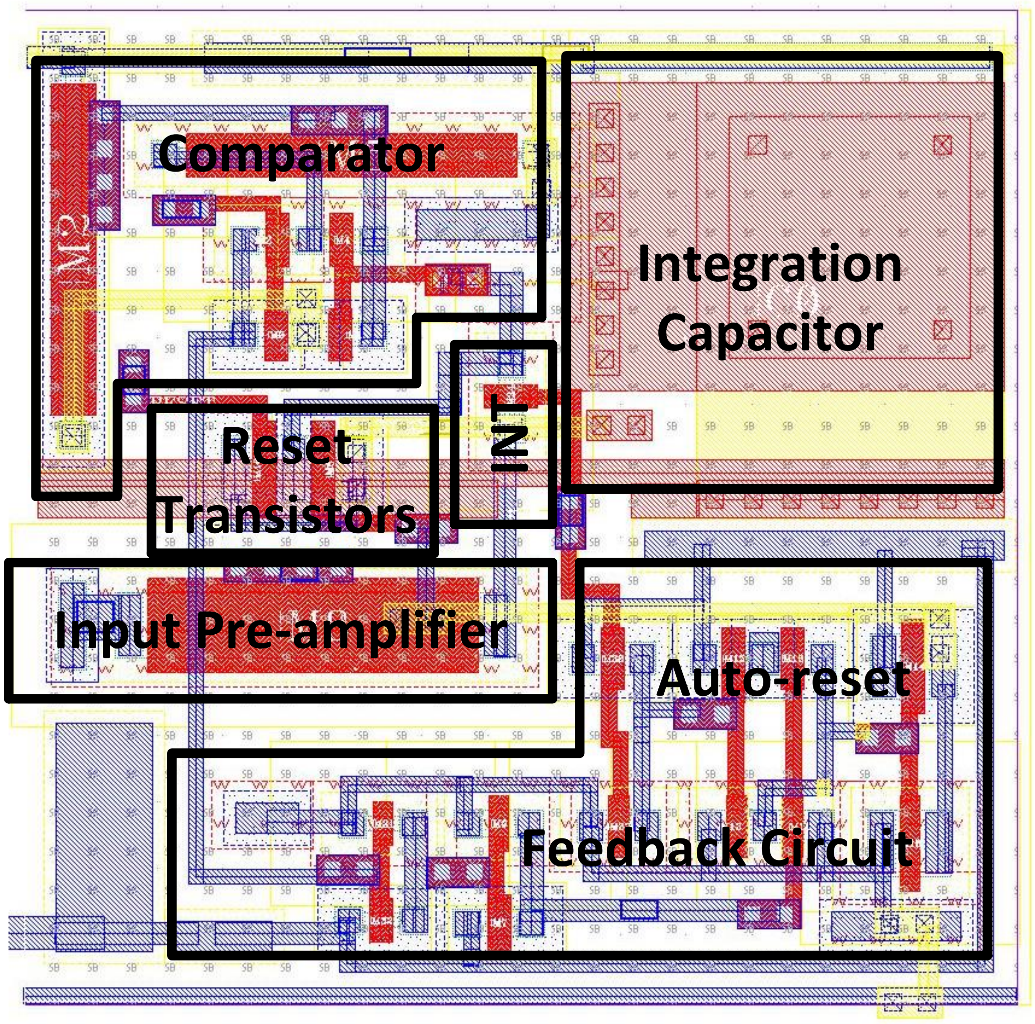


Figure 21: Pixel layout of the proposed digital TDI ROIC, dimensions: 15 μm X 15 μm

3.3 Ramp Generator

The ramp generator supplies the reference voltage (V_{ref}) to the comparators in all pixels both in the integration phase and the residual phase. Two signals (RESIDUE and RST_RAMP) control the ramp generator to generate the desired ramp signal. Since the comparator operates between 1 V and 1.75 V, the ramp generator output swing is also in this interval. Another important design parameter for the ramp generator is the driving capability of 10 pF in order to drive all pixels. It should also operate at cryogenic temperatures (77K).

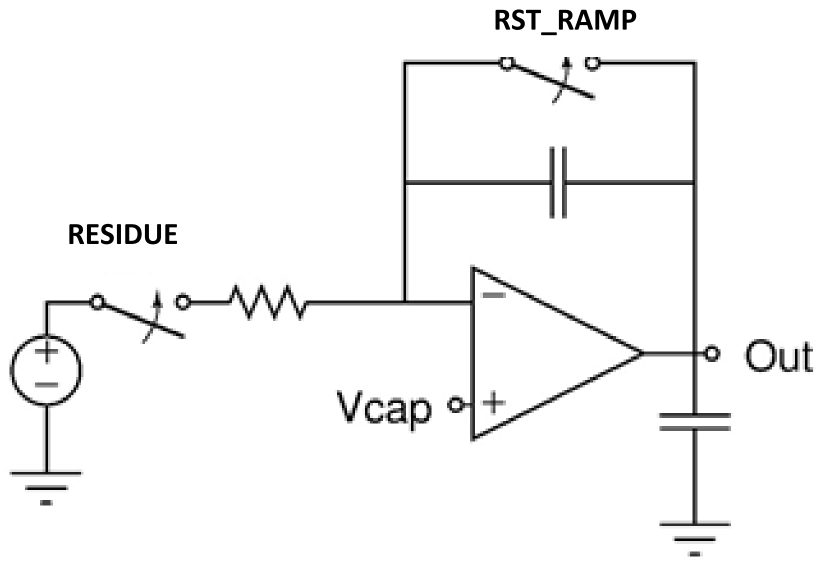


Figure 22: Ramp generator architecture

A switched capacitor integrator op-amp is used for the ramp generator. *Figure 22* shows the schematic of the ramp generator. During the integration phase RESIDUE switch is open, thus the integrated voltage on the feedback path of the op-amp is kept constant. When integration phase ends, the reset switch is activated and the output voltage of the ramp generator is same as the integration capacitor reset voltage (V_{cap}). This is to ensure that the ramp generator starts sweeping the dynamic range of the pixel comparator from the reset voltage of the comparator. The reset switch is active for a couple of clock cycles so that the output of ramp generator is stabilized at V_{cap}. After reset operation, RESIDUE switch is activated and the residual phase starts. An external bias is connected to a 23 kΩ resistance. The voltage difference between the external bias and V_{cap} generates a current flow, which is integrated on the feedback capacitor (5.5 pF) of the op-amp. The sizes of the resistance and the capacitance of the ramp generator are decided considering the desired output swing in 256 clock cycles during the residual phase of the TDI operation. The ramp generator should supply 0.7 V swing in 256 clock cycles with a reasonable power consumption. The power consumption of the ramp generator is 2 mW with 2.5 V power supply.

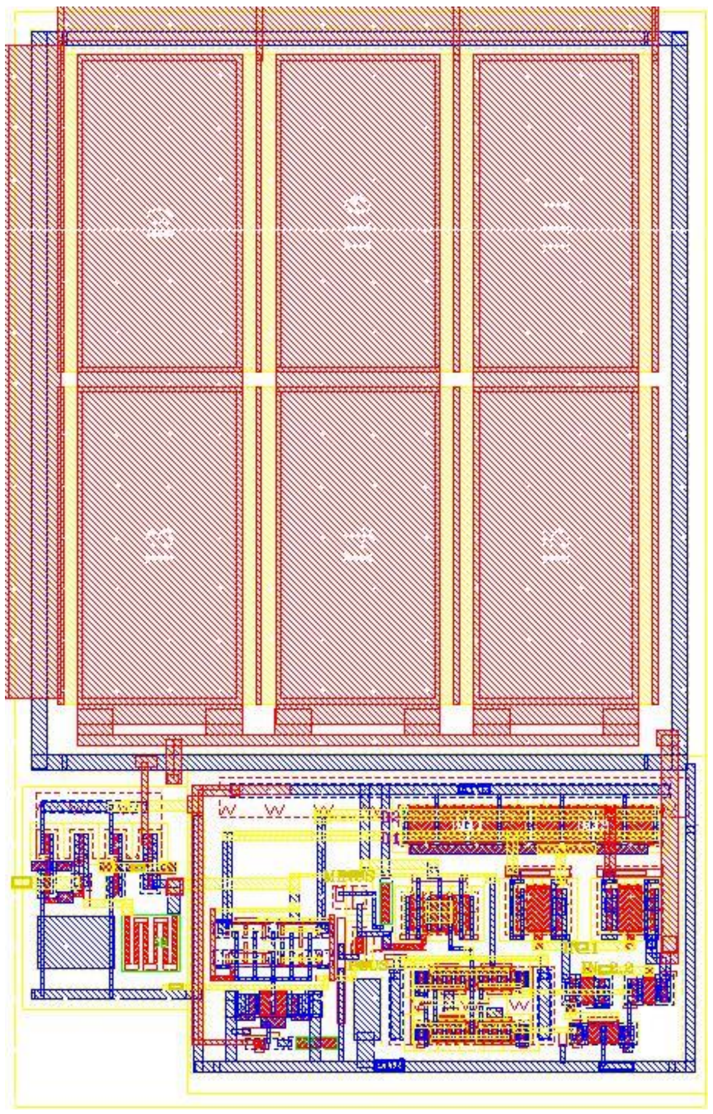


Figure 23: Ramp generator layout, dimensions: 65 µm X 105 µm

The ramp generator also gives the flexibility of changing the output swing and the residual phase time. By controlling the INT and the reset switch timings and adjusting the external bias, it is possible to change the residual phase duration and the output swing. Since both the ramp signal in the residual phase and the reference voltage in the integration phase are generated by the same circuit, there is no synchronization problem between these two signals. The layout of the ramp generator is shown in *Figure 23*.

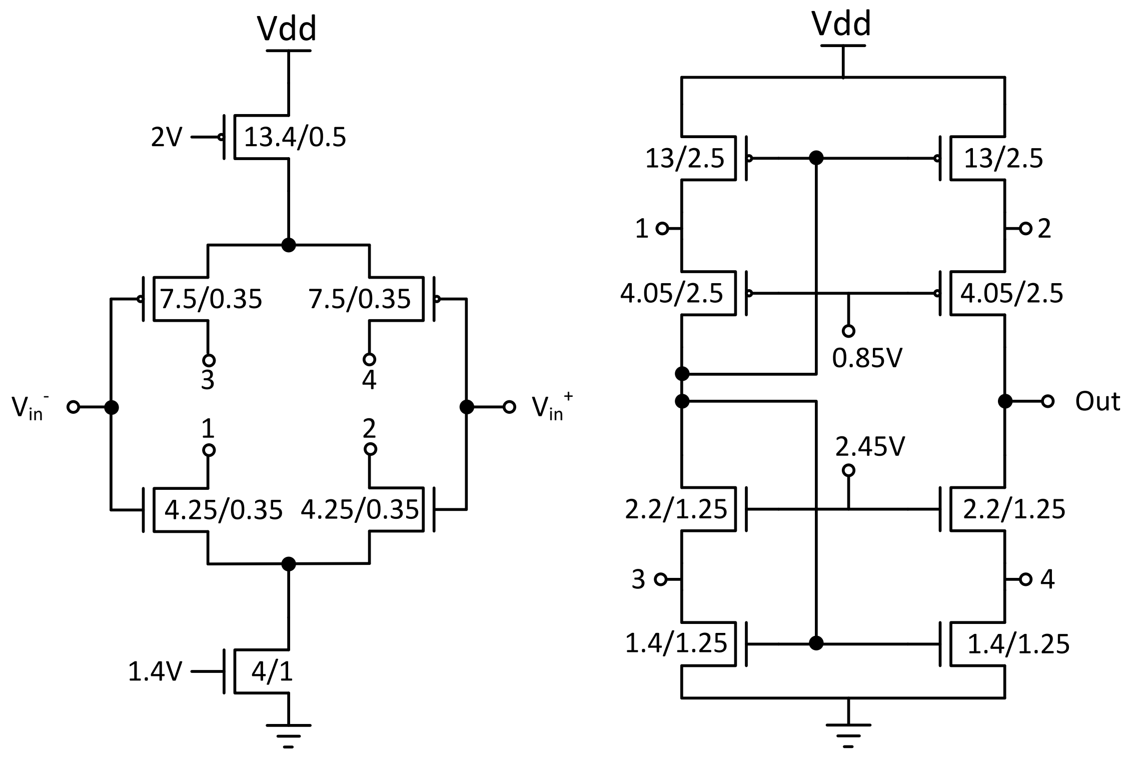


Figure 24: Op-amp schematic design for the ramp generator, all transistor sizes in μm

The circuit topology for the op-amp used in the ramp generator is complementary differential folded cascode amplifier topology. *Figure 24* shows the schematic design of the designed op-amp for the ramp generator. The nodes numbered 1,2,3 and 4 at the left side of the figure are connected to the same nodes at the right side of the figure. Complementary input pairs are seen as V_{in}^{+} and V_{in}^{-} at the left side of the figure. Complementary means input devices for the op-amp are both NMOS and PMOS devices. Complementary pairs are generally used to increase the input swing of the operational amplifier. For low input voltages, PMOS pairs are active whereas for high input voltages NMOS pairs are active. However, in this work, the reason to use the complementary pairs is to double the gain of the op-amp. Within the desired operating range of 1-2 V, both NMOS and PMOS pairs are active and provide gain, hence doubling the g_m of the amplifier assuming that g_ms of the NMOS and PMOS transistors are same. This is required to reach 75 dB gain with a single stage topology. Folded cascode technique is another technique that is commonly used for buffers. Instead of using the same type of MOS transistor as a bias transistor, different type of bias transistor is used. With a proper choice of bias voltages, it is possible to have a decent input and output swing with folded cascode amplifier. The op-amp is designed to operate at both room temperature (300 K) and cryogenic temperature (77 K), which is the main challenge of the op-amp design. In order to operate the op-amp both at room and cryogenic temperature, a PTAT and a CTAT circuit are added to the design. These PTAT and CTAT circuits compensate the threshold changes related to the operating temperature change. *Figure 25* shows the CTAT and PTAT circuits that are compensating the operating temperature change and generating bias voltages for the tail currents of ramp generator op-amp input pairs.

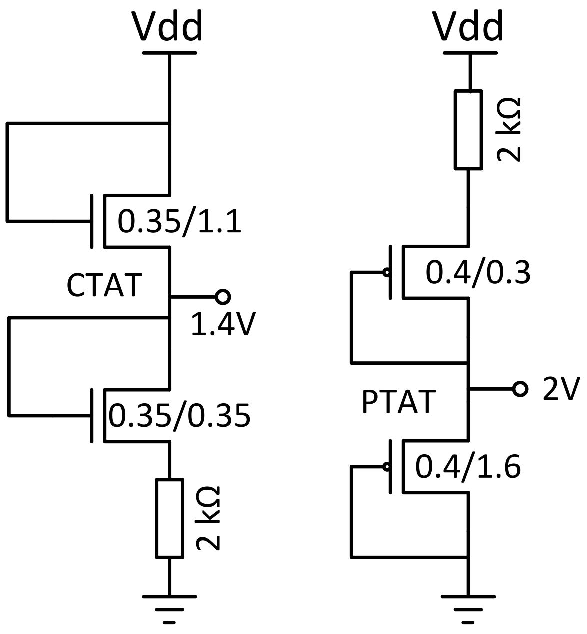


Figure 25: CTAT and PTAT circuits for the ramp generator

3.4 MSB and LSB Counter

Ripple carry counter is selected both as an MSB counter and LSB counter due to its area efficient architecture. Although Gray counter is better in terms of power consumption due to single transition per input trigger, ripple carry counter occupies a smaller area and also meets the speed requirement. Simulations show that with an input current of 50 nA together with 0.7 V swing and 40 fF integration capacitor, the comparator trigger frequency is around 1.8

MHz, which is important for the MSB counter. For the LSB counter, the maximum clock frequency is 250 MHz. Ripple carry counter can easily operate up to 1 GHz in the process node (180 nm CMOS) used for this prototype, which over-satisfies the speed requirement. *Figure 26* shows the ripple carry counter consisting of D flip flops (DFF) and a multiplexer to select which pixel's contribution is to be written. The multiplexer is controlled by the control circuit and has an enable signal that is activated when this counter is to be written in a frame. The output of one DFF is connected to the clock input of the next DFF. Negative outputs of each DFF are connected to their data input. The first DFF accepts the comparator trigger from its clock input, which is connected to the output of the multiplexer. *Figure 27* shows the layout of the 11-bit ripple carry counter for two rows. Two rows of the ripple carry counters are back to back connected (abutted) to gain space for the layout. Another circuit used in the ripple carry counter is the driving buffers. Since there are 32 counters (16 MSB + 16 LSB) each of which is 11-bit, it makes a long routing path for multiplexer inputs. Although there are driving buffers next to the pixels, it is required to put additional buffers into the middle of the counter rows to drive the multiplexer inputs without any problem. The drivers used for this purpose are conventional cascaded inverters with bigger transistor sizes.

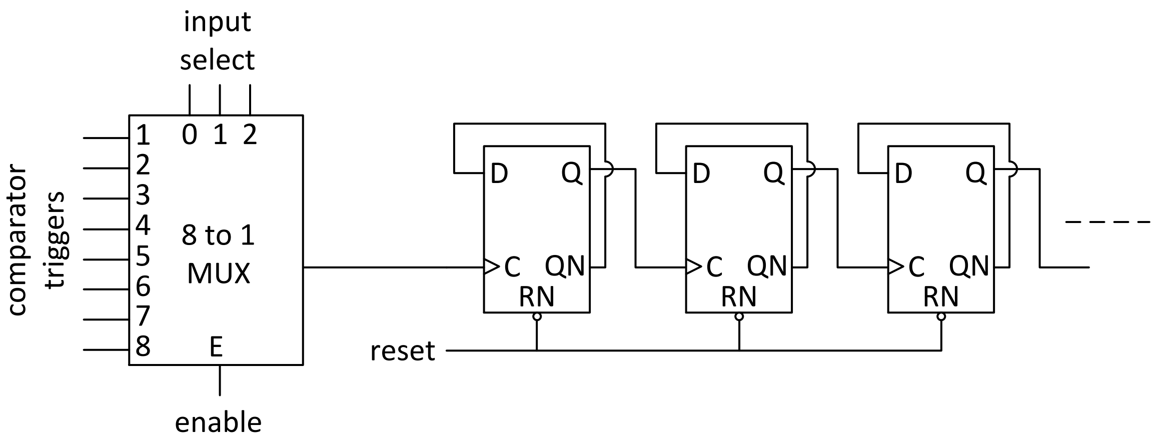


Figure 26: Ripple carry counter architecture used both for the MSB and LSB counters

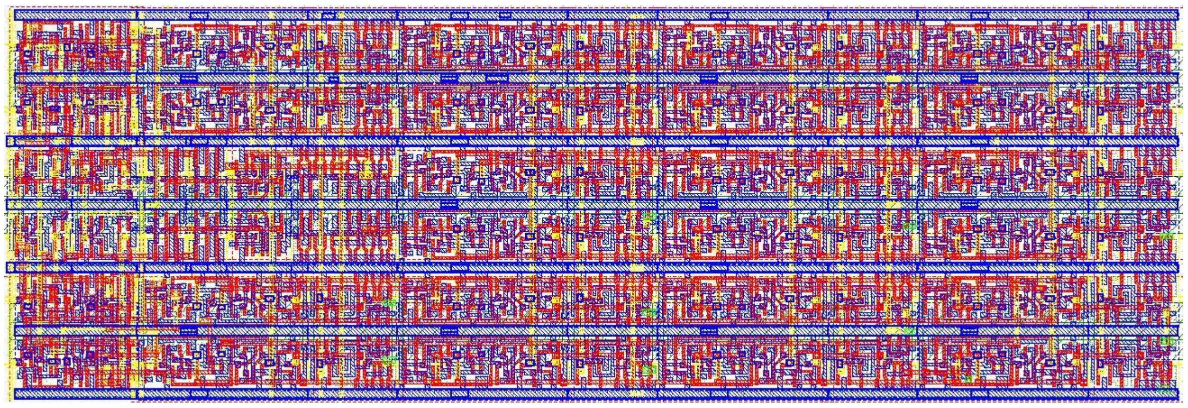


Figure 27: Layout of 11-bit ripple carry counter for two rows, dimensions: 90 μm X 30 μm

3.5 Control Circuit

The control circuit is another critical block that is controlling the whole ROIC. It takes 15 inputs and outputs 247 different signals. All the read, write, reset and multiplexing operations of 32 counters are controlled by the control circuit. TDI scan direction, bypass switches, and pixel deselection switches are also controlled by the control circuit. The control circuit also takes the charge of row multiplexing, ramp generator control signals, pixel control signals and serializer control signals. *Table 7* shows the input and output ports of the control circuit together with their functionality.

The control circuit is implemented using Verilog HDL language. Implemented circuit is verified through simulation results using Cadence NC Verilog / NC Sim tool. The control circuit is synthesized using Synopsys Design Compiler and finally, Cadence Encounter is used for the placement and routing of the synthesized netlist. The design is mapped to XFAB 180 nm junction isolated, low leakage standard cell library. *Figure 28* shows the auto-generated layout of the control circuit. The reason why the layout of the control circuit is so long, is the pin matchings of the control circuit and the counter rows.

Table 7: Control circuit input and output signals

Signal Name	**Direction**	**Functionality**
CLK	INPUT	Master clock
RST	INPUT	Master reset of ROIC
ENABLE	INPUT	Enables programming the control register of the ROIC
SERIAL_IN	INPUT	Serial input for serial programming
SERIAL_TRANS	INPUT	Enables serial programming
PAR_DATA<9:0>	INPUT	Inputs for parallel programming, bypass/TDI direction selection and pre-determined integration times
INT	OUTPUT	Integration time control for pixels
BP_SEL<7:0>	OUTPUT	Controls bypass switches for each column of pixels
TDI_SEL	OUTPUT	Controls TDI scan direction
SC_RST_MSB<15:0>	OUTPUT	Controls the reset operation of MSB counters
SC_RST_LSB<15:0>	OUTPUT	Controls the reset operation of LSB counters
MSB_ENABLE<15:0>	OUTPUT	Controls the enable inputs of the multiplexers in the MSB counters
LSB_ENABLE<15:0>	OUTPUT	Controls the enable inputs of the multiplexers in the LSB counters
SC_MSB<2:0><15:0>	OUTPUT	Controls the selection inputs of the multiplexers in the MSB counters
SC_LSB<2:0><15:0>	OUTPUT	Controls the selection inputs of the multiplexers in the LSB counters
RST_CAP	OUTPUT	Controls the reset operation of the integration capacitors in pixels
RST_RAMP	OUTPUT	Controls the reset operation of the ramp generator
LOAD	OUTPUT	Controls the load of 22 bits from MSB and LSB counters’ output bus to serializer
RESIDUE	OUTPUT	Controls the ramp generator switch required to sweep the ramp signal
SC_READ_MSB<15:0>	OUTPUT	Controls the read operation of MSB counters
SC_READ_LSB<15:0>	OUTPUT	Controls the read operation of LSB counters
ROW_SELECT<89:0>	OUTPUT	Controls the multiplexing of rows to be connected to the serializer

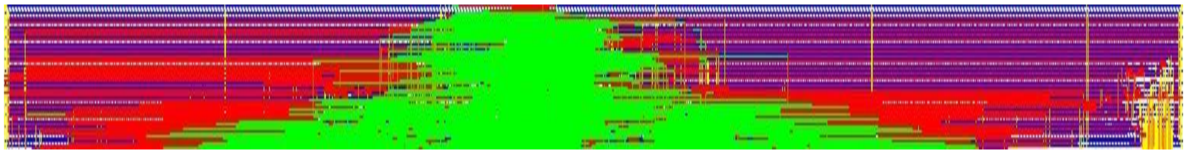


Figure 28: Control circuit layout, dimensions: 3280µm X 125µm

3.6 Serializer

The serializer is used to convert the parallel 22-bit data (11-bit MSB + 11-bit LSB) to a single bit. The main purpose of the serializer is to overcome the pad limitation problem. The serializer is implemented using Verilog HDL language, simulated using NC Verilog, synthesized using Synopsys Design Compiler and placed & routed using Cadence Encounter similar to the control circuit. The same standard cell library with the control circuit is used, namely XFAB 180 nm junction isolated, low leakage library.

Table *8* shows the input/output signals of the serializer and Figure 29 shows the layout of the serializer.

Table 8: Input/output signals of the serializer

Signal Name	Direction	Functionality
CLK	INPUT	Master clock same with the control circuit
RST	INPUT	Master reset same with the control circuit
ENABLE	INPUT	Controls the load of 22-bit from MSB and LSB counters' output bus to the serializer
PARALLEL_IN<21:0>	INPUT	Parallel 22-bit input data
SERIAL_OUT	OUTPUT	Single bit output data

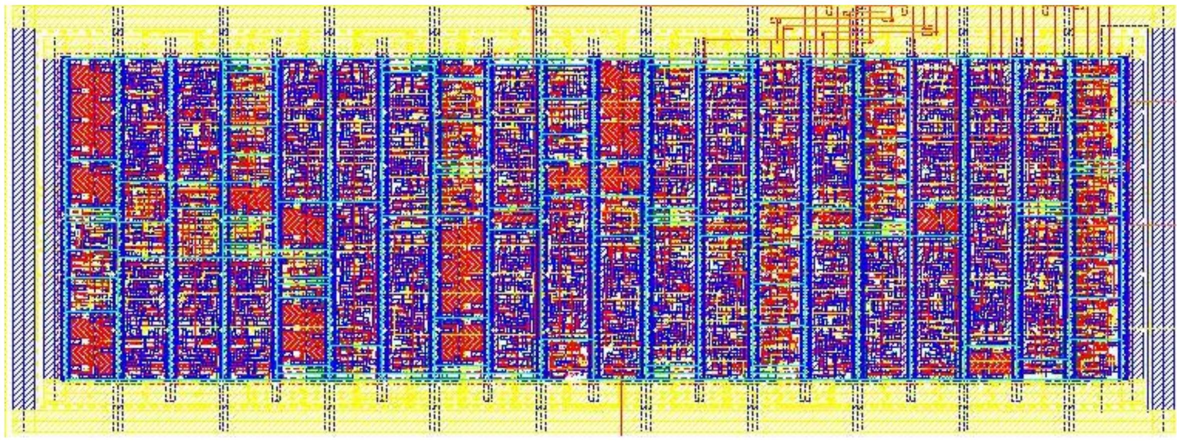


Figure 29: Layout of the serializer, dimensions: 110 μm X 40 μm

3.7 Test Current Source

In order to be able to test the designed ROIC without a real bonded detector, a test current source that is immune to process variations is designed [58]. At the beginning of the thesis work, a single PMOS transistor is used as a current source. However, it is observed that a single PMOS transistor is prone to process variations. For example, 59% variation in output current is observed for a 0.7 nA average current value and 34% for 34 nA. Current variation is very important in terms of the TDI implementation because it is assumed that all input currents are same for all TDI pixels. Any variation of current within 8 TDI pixel is reflected as a noise to the ROIC. To solve this problem, a current source that can tolerate the threshold voltage changes due to process variations is designed.

Figure 30 shows the designed current source. Transistors M_1 and M_2 are chosen in a way that, they generate a V_X voltage independent of V_{th} variations of the transistors, assuming that these transistors are in close proximity and exposed to the same process gradient. M_3 is basically a current source that is controlled by a process-invariant gate voltage. Finally, M_4 is used for shielding purposes. *Figure 31* shows the layout of the process invariant current source. The height of the current source is same as the height of the pixel (15 μm) to fit into the same row.

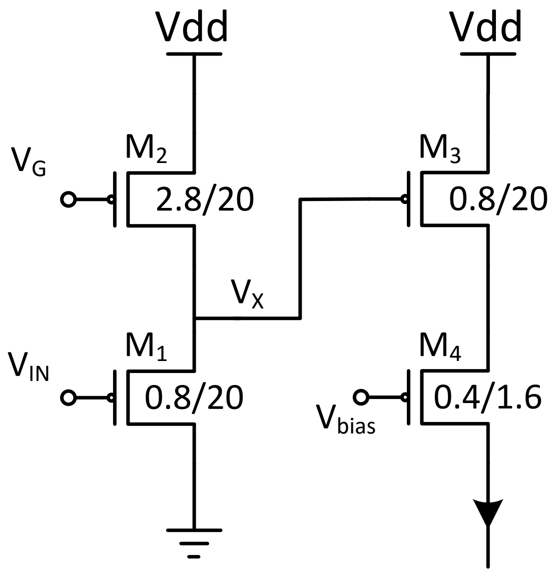


Figure 30: Schematic of the current source that is immune to process variations, transistor sizes in μm

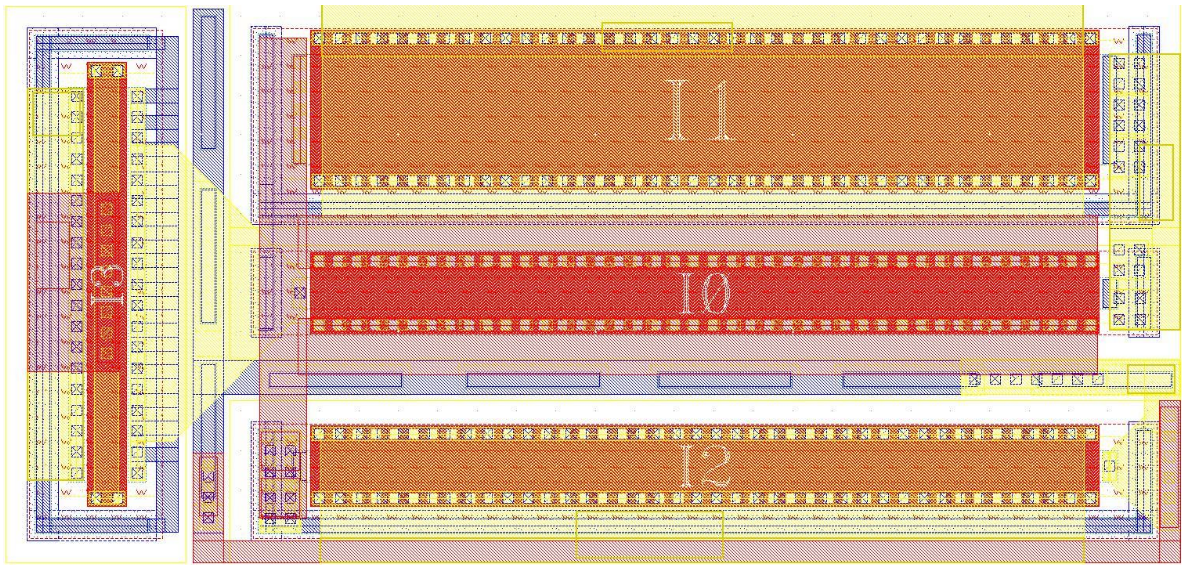


Figure 31: Layout of the test current source, dimensions: 30 μm X 15 μm

3.8 Layout of the Prototype

The prototype of the proposed digital TDI ROIC is implemented with XFAB 0.18 μm XH018 CMOS process. The main reasons why this process technology is preferred are the availability of the technology in the long term, fabrication cost, process options of XH018 such as multiple threshold MOS devices, isolated well devices, wide range and well characterized standard cell libraries. It was possible to implement the same design with 0.35μm technology, but it will not be available for a long time and most importantly there is no need for high voltage process nodes to increase the dynamic range of pixel as opposed to the analog TDI ROICs. It was also possible to use low voltage process nodes such as 90 nm CMOS technology, that come with low power consumption benefit. However, using advanced CMOS processes is also not efficient in terms of fabrication cost.

Figure 32 shows the layout of the prototype 90x8 array for digital TDI ROIC, which occupies an area of 4.6 mm X 3 mm. 128 I/O pads are used for the bias voltages, supplies, digital control signals, digital and analog outputs of the prototype 90x8 array and related test blocks. Thus, the prototype is a pad limited design. Most of the pads are used for digital outputs verifying the functionality of various blocks.

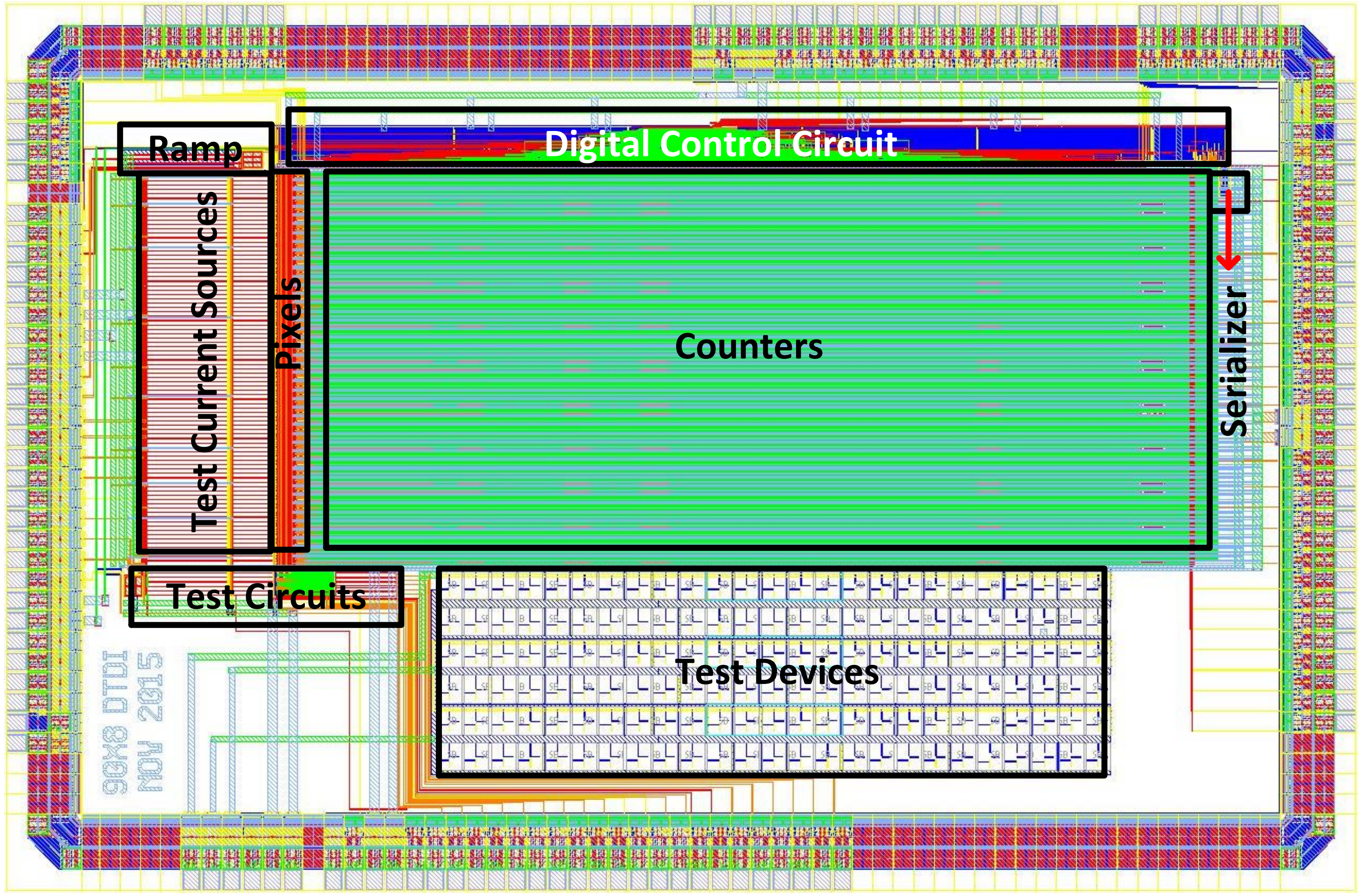


Figure 32: Layout of the prototype 90x8 digital TDI ROIC, dimensions 4.6 mm X 3 mm

4. SIMULATION and MEASUREMENT RESULTS

4.1 Simulation Results

Verification of various sub-blocks and 90x8 digital TDI ROIC array are done through simulations. Block level simulations are done with Cadence Spectre with conservative settings both in schematic level and post-layout level. Simulations of the control circuit and the serializer are done with Cadence NC SIM Verilog simulator. Synthesized netlists of these auto-generated circuits are also simulated with Cadence Spectre with liberal setting just in schematic level. Since the synthesized digital circuits include many devices and incorporate clock signal, post-layout simulations of these circuits take very long time. Even behavioral level simulations of the auto-generated circuits seem to be satisfying for low-speed clocks.

System level simulations of the prototype array are done with mixed signal simulation tools of Cadence with the multi-threading option. Mixed-signal simulation runs the control circuit and the serializer in behavioral mode, counters in schematic mode while running the pixels and the ramp generator in post-layout extracted views. Another important point for the array simulation is the circuit size used in the mixed signal simulation. It is nearly impossible to run a simulation with the whole array because it takes very long time. Due to this computation limitation, 2 consecutive rows are included in the simulation to reduce the circuit complexity. Another version of this simulation, that includes a row from the top of the array and a row from the bottom of the array, is also performed. In this way, it is possible to verify all the aspects and functionalities of the proposed digital TDI architecture.

4.1.1 Pixel Simulations

Various parameters of the pixel are verified through simulations. The first parameter is the linearity of the pixel with respect to varying input current. For this simulation, input current value is swept while keeping the integration time fixed. For each input current, comparator auto-triggers are counted. *Figure 33* shows the linearity graph of the pixel.

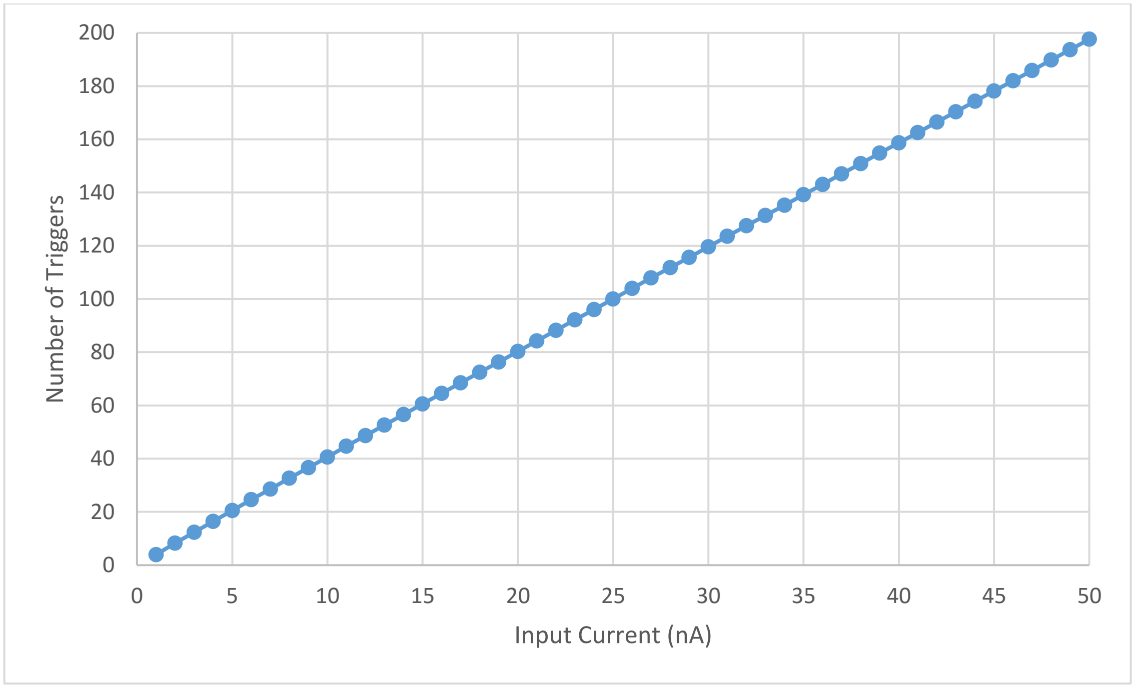


Figure 33: Linearity simulation of the pixel with input current between 1 nA and 50 nA

The second parameter simulated for the pixel is the uniformity parameter. Uniformity between the pixels in the same row is very crucial for TDI since all pixels add their contribution to the final value. Any variation between the pixels is reflected as an additional noise to the output. In schematic simulations, all pixels switch at the same time with the same inputs and control signals given. However, in the post-layout simulations, it is observed that the first pixel in the same row behaves differently. To solve this problem, a dummy pixel, that is not connected to the counters but operates as same as the other counters, is inserted into the row as the first pixel. *Figure 34* shows the post-layout simulation result of all 9 pixels including the dummy pixel. When the integration capacitor voltages of the pixels reach the reference value, the comparator is auto-triggered. While all 8 pixels switch within a very narrow interval around 23.7 µs, the dummy pixel switches very late from the other pixels around 24.1 µs. Since the dummy pixel is not used in TDI summation counters, it does not create any problem, rather it solves the uniformity problem of the first pixel.

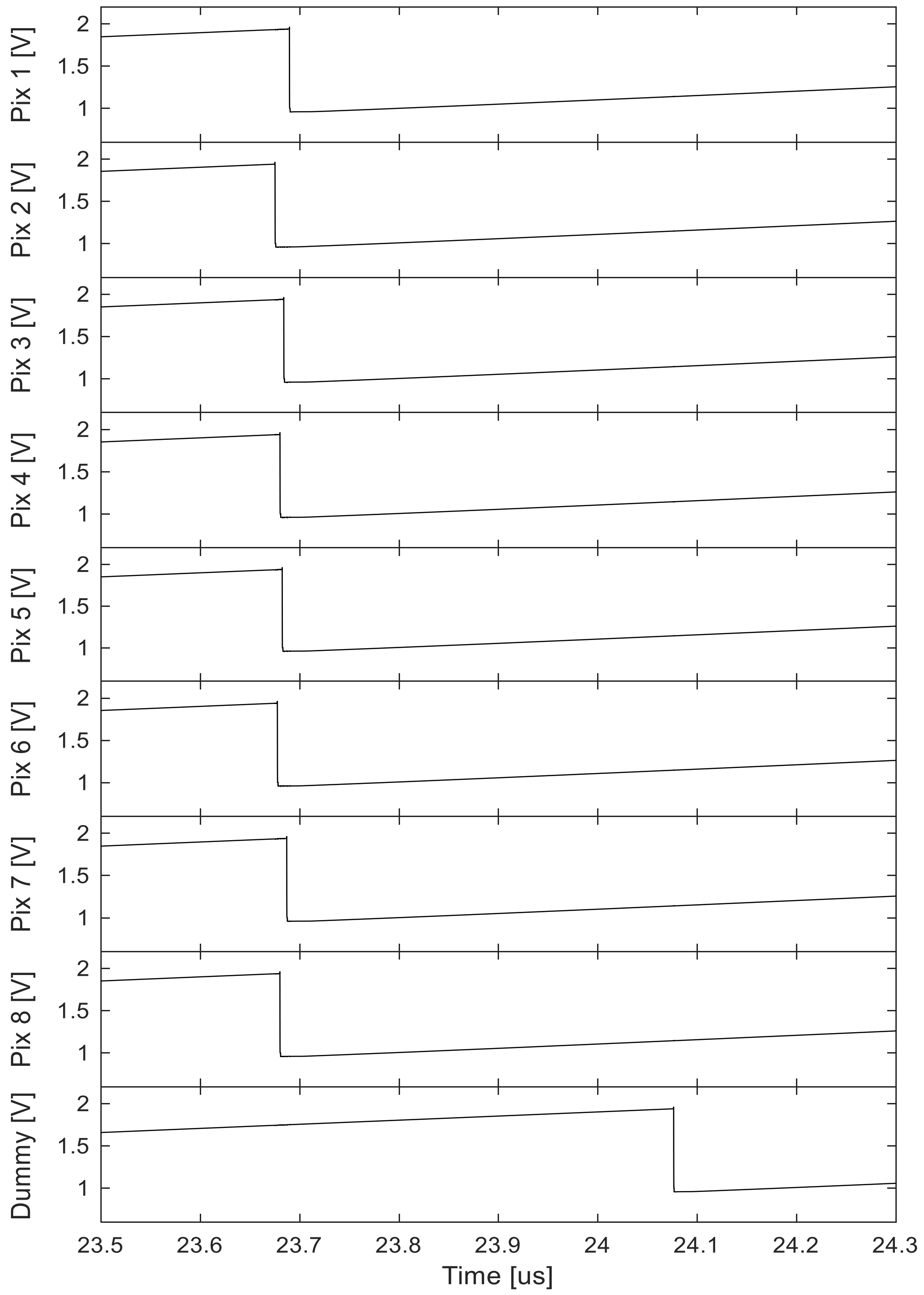


Figure 34: Post-layout simulation of all 8 pixels in a row with an additional dummy pixel

4.1.2 Ramp Generator

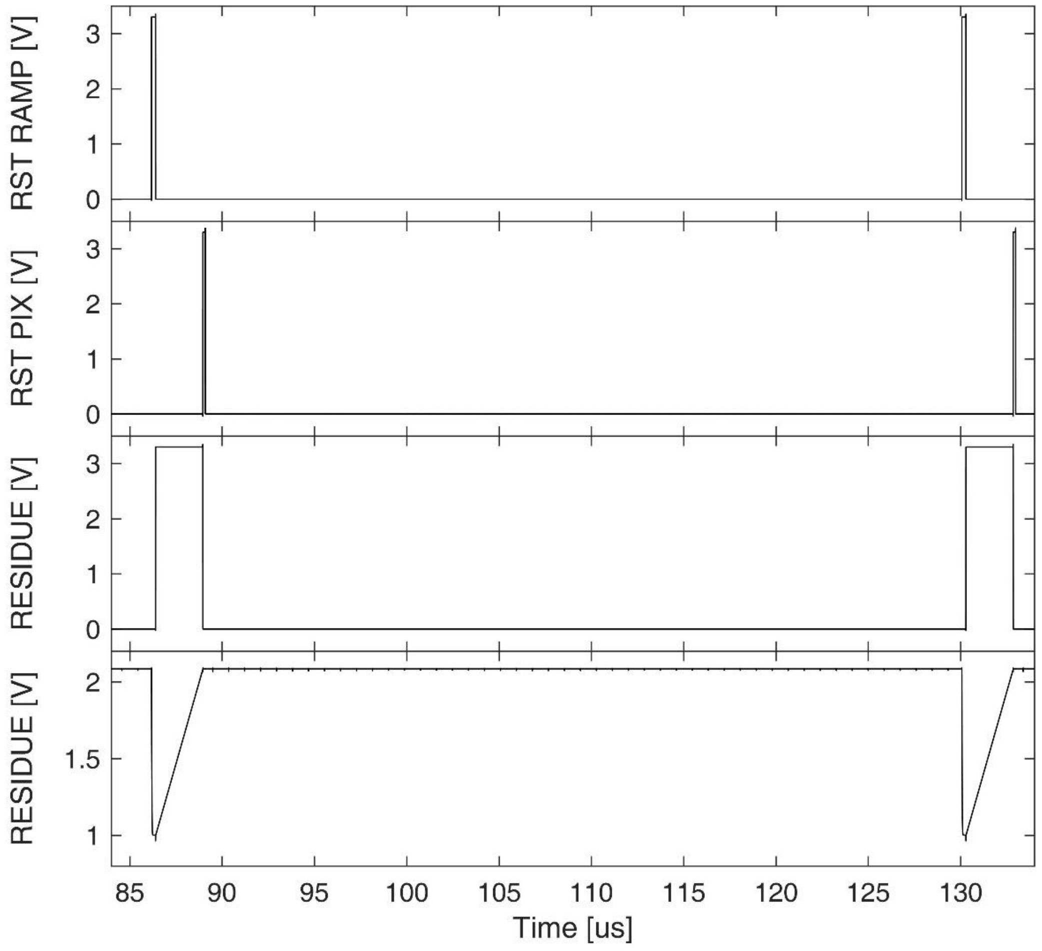


Figure 35: Post-layout simulation result of the ramp generator

Figure 35 shows the post-layout simulation results of the ramp generator. The ramp generator is responsible for generating the fixed reference voltage during the integration phase and the ramp signal during the residual phase. As the simulation results show, when RST RAMP is activated, the ramp generator is reset, and the output of the ramp generator (VREF) goes to the capacitor reset voltage level. Then RST RAMP is deactivated, and RESIDUE is activated. In the residual phase of the operation, ramp generator output goes from the capacitor reset voltage level to the reference voltage level, sweeps the whole dynamic range. If both RST

RAMP and RESIDUE signals are not active, the ramp generator output stays constant at the reference voltage level. As the simulation validates, the ramp generator supplies the reference signal to the pixel comparators both in the residual phase and integration phase successfully.

4.1.3 Counters

Figure 36 shows the MSB counter output after the contribution of the first pixel. PIX OUT shows triggers of the pixel comparator. Each time the comparator is triggered, the output count increments by one. At the end of the integration time, just the first-pixel contribution is written to the counters and the content of the MSB counter shows 10 (0-0-0-0-0-0-0-1-0-1-0). *Figure 37* shows the simulation result of the LSB counter while the first pixel is connected to this counter. PIX OUT signal, which is the signal at the bottom, is connected to the enable input of the LSB counter during the residual phase of the operation. While PIX OUT is high, LSB counter counts the number of clock cycles. When PIX OUT is low, the LSB counter stops counting the clock cycles. The other top 11 signals represent the 11-bit outputs of the LSB counter. The content of the LSB counter shows 181 (0-0-0-1-0-1-1-0-1-0-1). It is worth to mention that these MSB and LSB counter outputs show the contribution of just a single pixel. The final content in these counters will be the total count of all 8 pixels, which is then divided by 8 for averaging the signal to improve SNR of the final image.

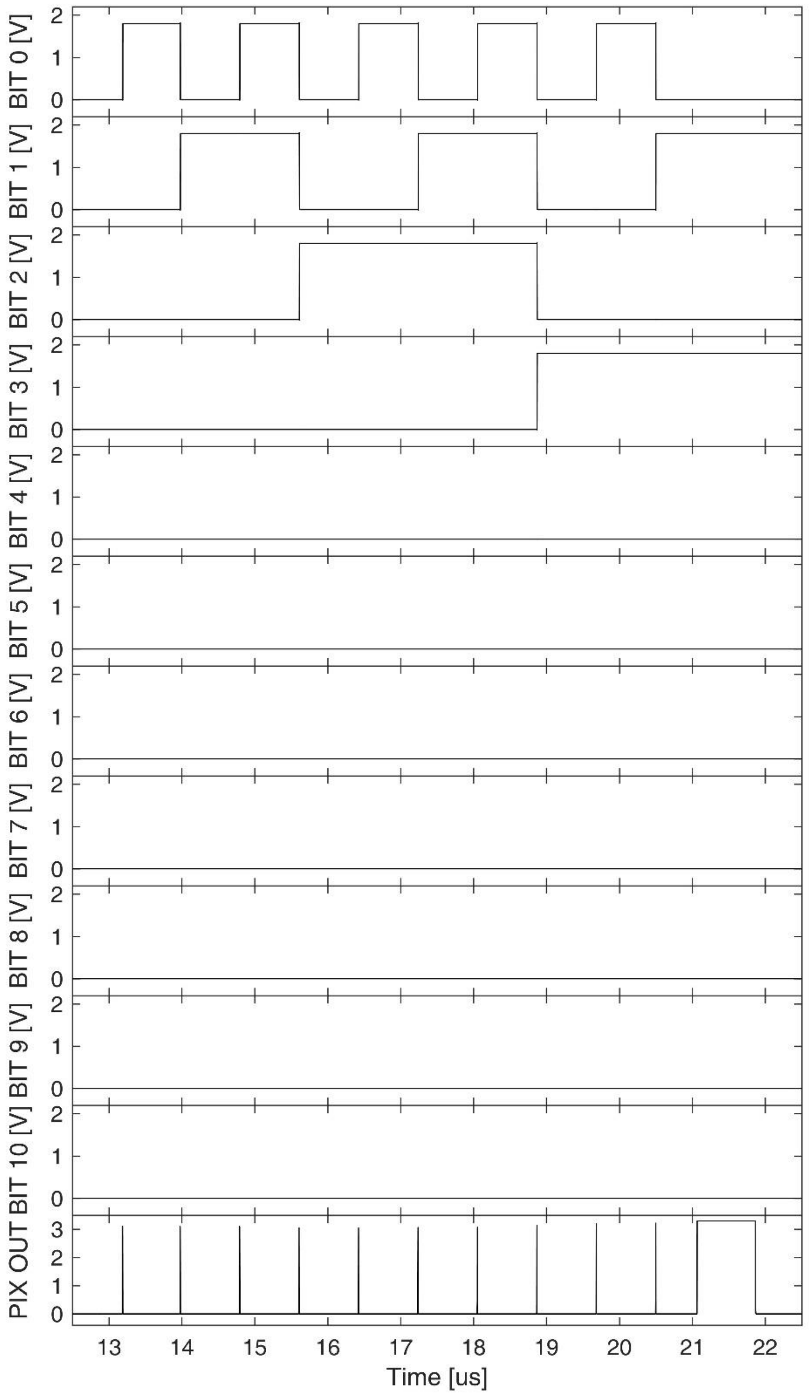


Figure 36: Simulation result showing MSB counter output after the contribution of the first pixel

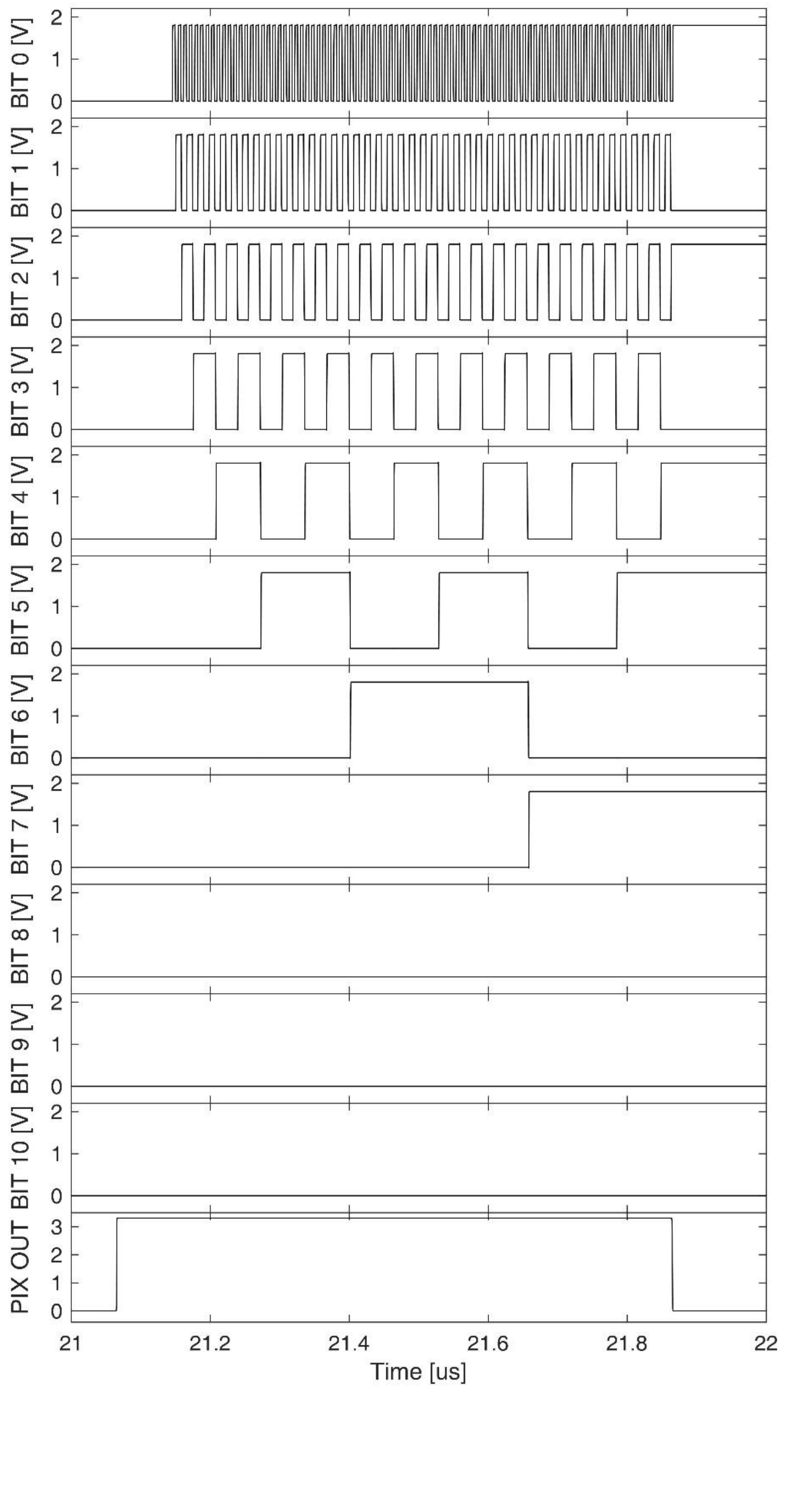


Figure 37: Simulation result of the LSB counter after the contribution of the first pixel

4.1.4 Control Circuit

All the simulation results that are presented in this section of the thesis except the pixel and current source simulations are recorded under the control of the control circuit. These simulations are recorded from the different blocks of the system level simulation. All these simulations containing the outputs of the control circuit shows that the control circuit perfectly generates the control signals required for the pixel, the ramp generator, the counters, the serializer and all switches.

It is not possible to include all control signals in the simulation graph. However, Figure 38 shows the control signals of first MSB and LSB counters as representative signals of the control circuit to give an idea for other signals. S<2:0> signals show the selection control signals of the multiplexers in the counters. According to the TDI algorithm that is shown in Table 6, the desired pixel is selected by the control circuit to write to the MSB and LSB counters. Simulation results show that in each two frames multiplexer selection signals change starting from the first pixel to the last pixel. In the first frame that the multiplexer inputs change, first MSB counter is enabled, which means comparator triggers in the selected pixel is written to the corresponding MSB counter. Following the MSB WR signal, LSB WR signal is activated, and the number of clock cycles that is proportional to the residual charge on the integration capacitor, is counted and written to the corresponding LSB counter. Due to the oversampling rate of 2, the next frame, these selected pixels do not write any data to any counter. After all pixels are selected and contribution of these pixels are written to the counters, the READ signal is activated and the content of the MSB and LSB counters are sent to the serializer. Following the READ signal, the RST signal is activated. Since the reset operation of the counters is active low, the RST signal goes to the low state to activate counter reset operation as the simulation results show. INT signal is put as a reference signal to understand the beginning and the end of the frames. Each frame is defined between two positive edges of the INT signal.

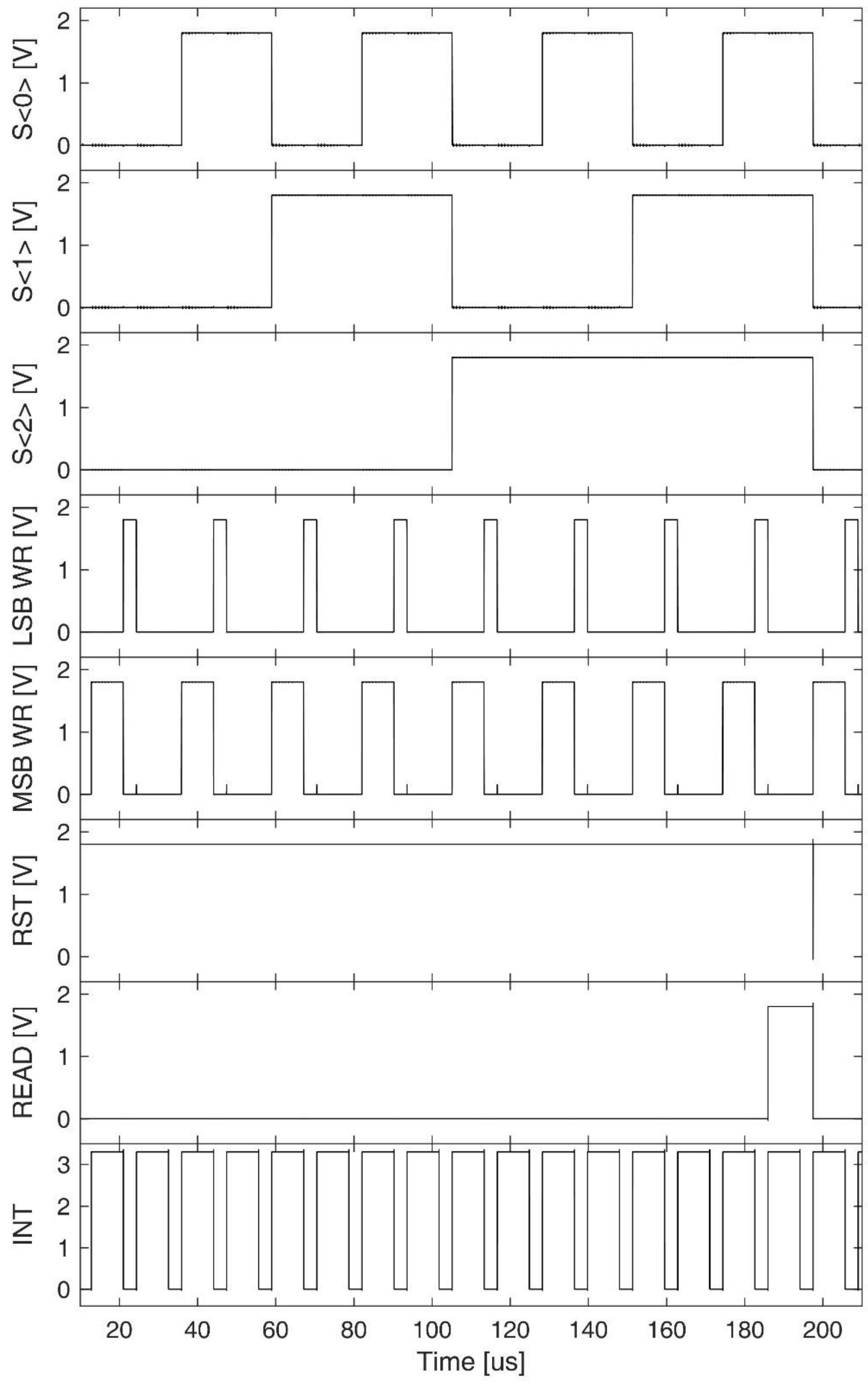


Figure 38: Simulation result showing control circuit outputs for the first MSB and LSB counters

4.1.5 Serializer

Figure 39 shows the output of the 90x8 prototype in synchronous with some control signals. At the beginning of the frame, the first row is activated for outputting its content. After the first row, the second row is activated. All 90 rows are selected one by one to output their content. The simulation shows the first two rows. When it is high, it means that the selected row of the array outputs its content to the bus to be captured by parallel to serial converter (serializer). The LOAD signal shows the time interval that the content at 22-bit output bus is captured by the serializer. When it is high, 22-bit output bus content is transferred to the serializer. SER OUT shows the output of the serializer. The first bit is the least significant bit of 22 bits. As it is seen from the figure, the content is "(1-1-1-1-0-1-0-1-1-0-1)-(0-0-0-0-1-0-1-0-0-0-0)" from the left to the right. The left parenthesis shows the LSB content while the second parenthesis shows the MSB content. So, the binary content of the MSB counter at the first row is "00001010000". Similarly, the binary content of the LSB counter at the first row is "10110101111". The MSB part shows 80 in decimal notation which is in line with our pixel simulation results in Section 4.1.7. As it is mentioned in Section 4.1.7, pixel comparator is triggered 10 times in a single frame and there are 8 TDI pixels, which makes 10x8=80. The LSB part shows 1455 in decimal notation which means 181 or 182 clock cycles are counted by the LSB counters for a single pixel in average. Average LSB is 181.875 meaning mostly the LSB counter is 182. The same output content is also valid for the second row due to the same amount of input photocurrent applied to all pixels in the first two rows.

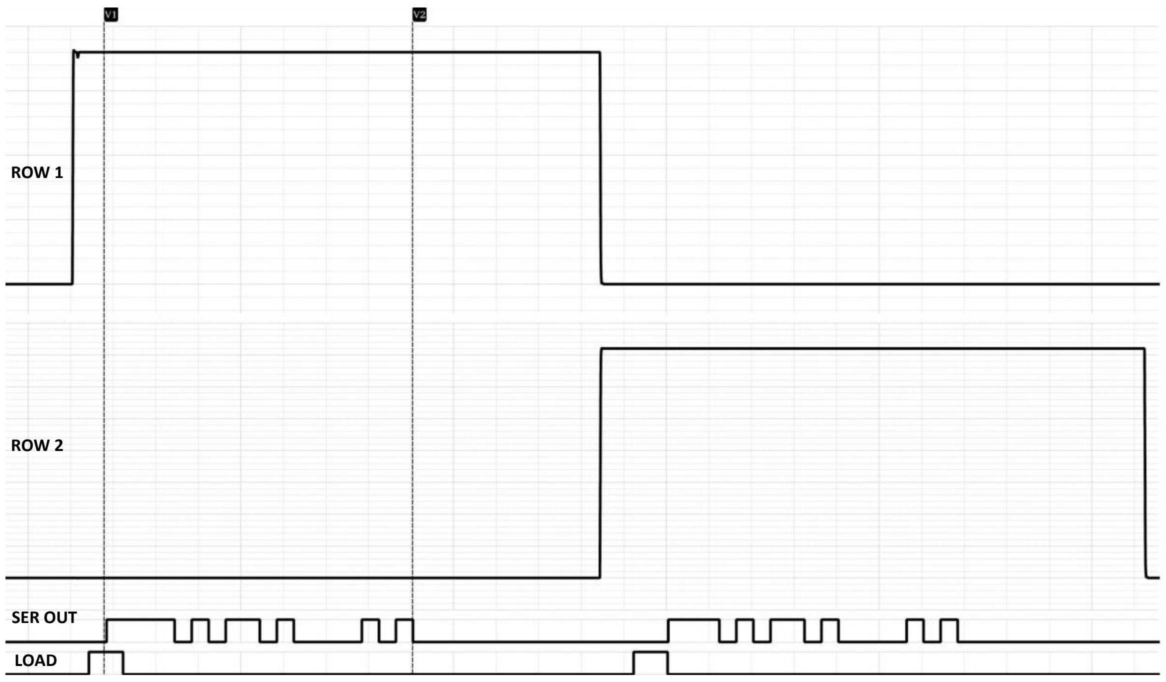


Figure 39: Simulation result showing the operation of the serializer

4.1.6 Current Source

Figure 40 shows the Monte Carlo simulation result of the current source for an input current of 1 nA. The current variation due to process and mismatch variations is simulated as 12%, which is a significant improvement compared to the single transistor current source which was 59% for the same amount of input current. Process variation compensation of the current source is worse for small currents. For higher currents, compensation of the current source is even better. *Figure 41* shows the simulation result for a higher input current of 10 nA. The variation of the current due to process and mismatch variations is simulated as 7%, which verifies that the current compensation works better for higher currents.

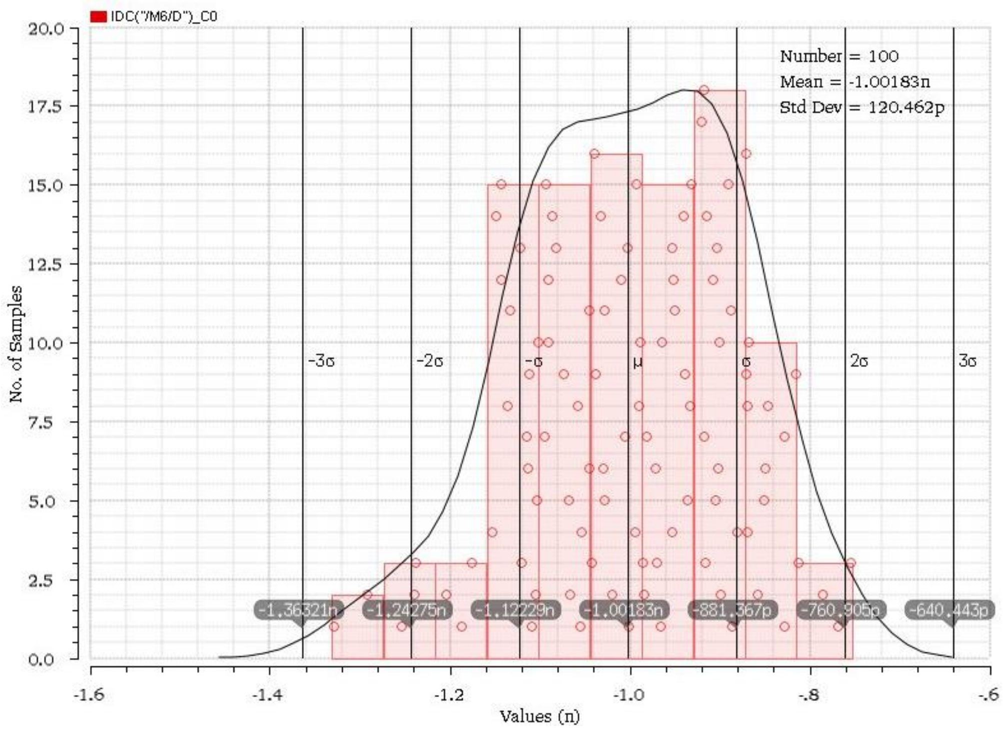


Figure 40: Monte Carlo simulation result of the current source for input current of 1 nA

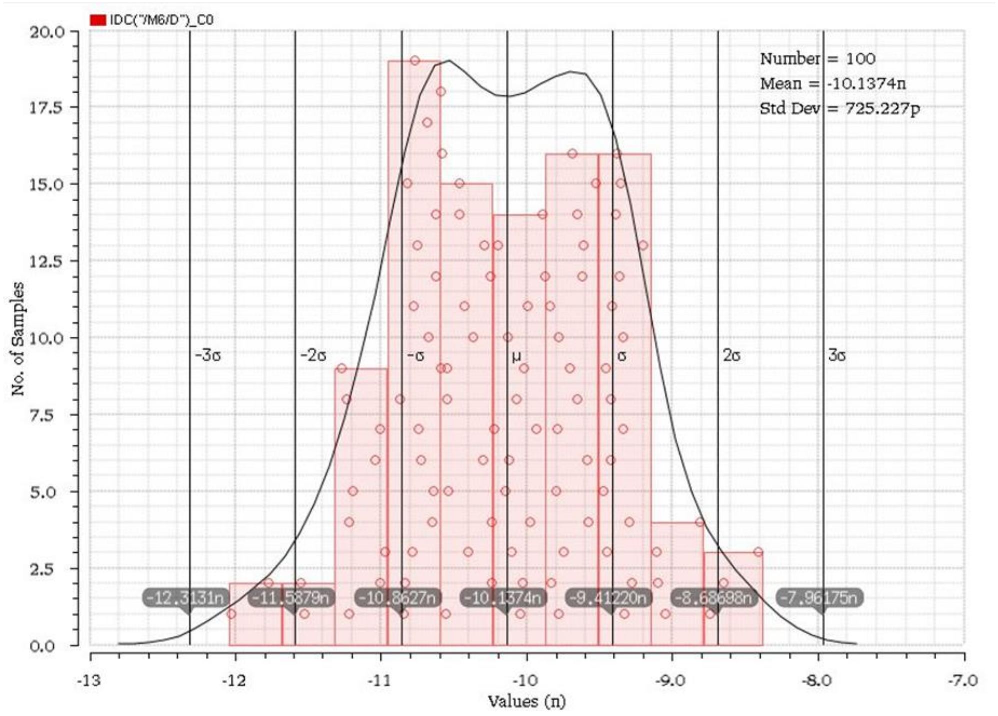


Figure 41: Monte Carlo simulation result of the current source for an input current of 10 nA

4.1.7 System Level Simulations

System level simulation results showing the pixel block of the prototype array are presented in this section. *Figure 42* shows the simulation result of a pixel. The integration signal (INT), which is controlling the integration time, is an output of the digital control circuit. Before the INT signal is activated, the voltage level at the integration capacitor node is at 1 V and the reference voltage is 2 V. When the INT signal is activated, the photocurrent is integrated onto the integration capacitor. When the integration capacitor node comes to VREF reference voltage, the comparator in pixel is triggered and the feedback circuit resets the integration capacitor. After that, the next integration starts. This trigger signal is the PIX OUT signal and is counted by the MSB counter in that row. Triggering and the counting operation continues until the end of the integration period. As it is seen, the comparator is triggered 10 times. When the integration period is over, there is a remaining voltage on the integration capacitor that is not quantized. The red graph shows the integration capacitor voltage (VPIX). If this residual voltage on integration capacitor is not quantized, this generates a large quantization noise. For the fine quantization operation, there is a residual phase that can count 256 clock cycles and generates the 8-bit output that is proportional to the residual voltage on the integration capacitor. In this phase of the TDI operation, the ramp generator sweeps the whole dynamic range of the pixel and the comparator waits for the ramp generator output voltage to reach the level of residual voltage on the integration capacitor. When ramp generator reaches that value, the comparator is triggered and the pixel digital output goes to the low level. The width of the pulse in residual phase is proportional to the amount of residual voltage on integration capacitor as mentioned earlier. The RST PIX signal is the integration capacitor reset signal that is used to reset that node at the beginning of each frame after the residual phase of the previous frame ends. The RST RAMP signal is used to reset the ramp generator at the beginning of the residual phase. The control circuit generates all these control signals.

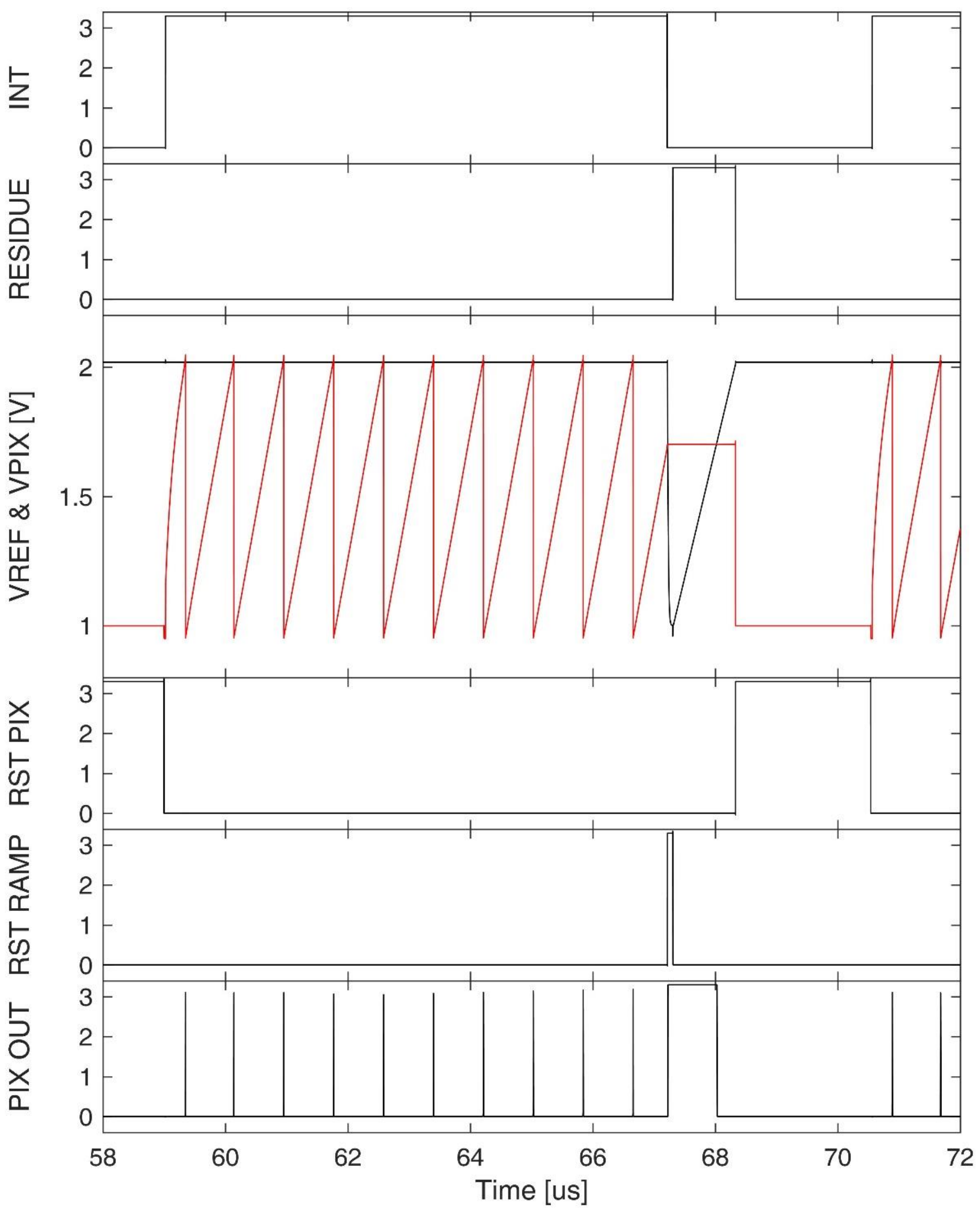


Figure 42: Pixel simulation result of the 90x8 digital TDI ROIC

4.2 Measurement Results

4.2.1 Measurement Setup

To verify the functionality and the performance parameters of the prototype 90x8 digital TDI ROIC, a low noise mixed-signal measurement setup is required. A testing equipment, named Pulse Instruments that is used for the testing of the imaging systems is used to supply all bias voltages, power supplies and digital control signals including the master clock. The prototype ROIC is operated with 100 MHz and 200 MHz clock signals. Test circuits are also measured to verify the functionality of the pixels, the ramp generator, a single row in order to be able to test the sub-blocks of the ROIC independent of the control circuit. For this purpose, a logic analyzer is used to supply the digital control signals and the master clock. The reason why Pulse Instruments is not utilized for the test blocks is the limitation of the number of digital control signals. Pulse Instruments can supply up to 8 digital control signals while test circuits need 13 different control signals in order to control the pixels, the ramp generator, the counters, and the serializer independently.

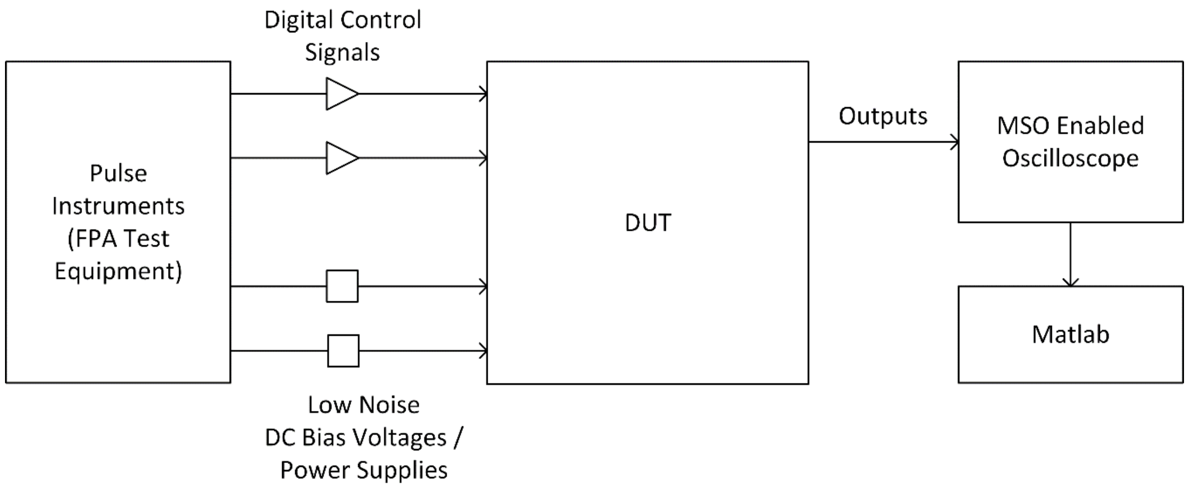

Digital Control Signals
Pulse Instruments (FPA Test Equipment)
DUT
Outputs
MSO Enabled Oscilloscope
Matlab
Low Noise DC Bias Voltages / Power Supplies

Figure 43: Measurement setup of the prototype digital TDI ROIC

The measurement setup is shown in *Figure 43*. The control signals and required dc supplies are connected to the device under test (DUT). Outputs of the DUT are captured with a mixed signal oscilloscope. The recorded data is processed in Matlab to calculate the noise and SNR of the DUT. *Figure 44* shows the photo of the measurement setup. Pulse Instruments FPA test equipment is not shown in the photo, since it is big equipment and located away from the MSO enabled oscilloscope.

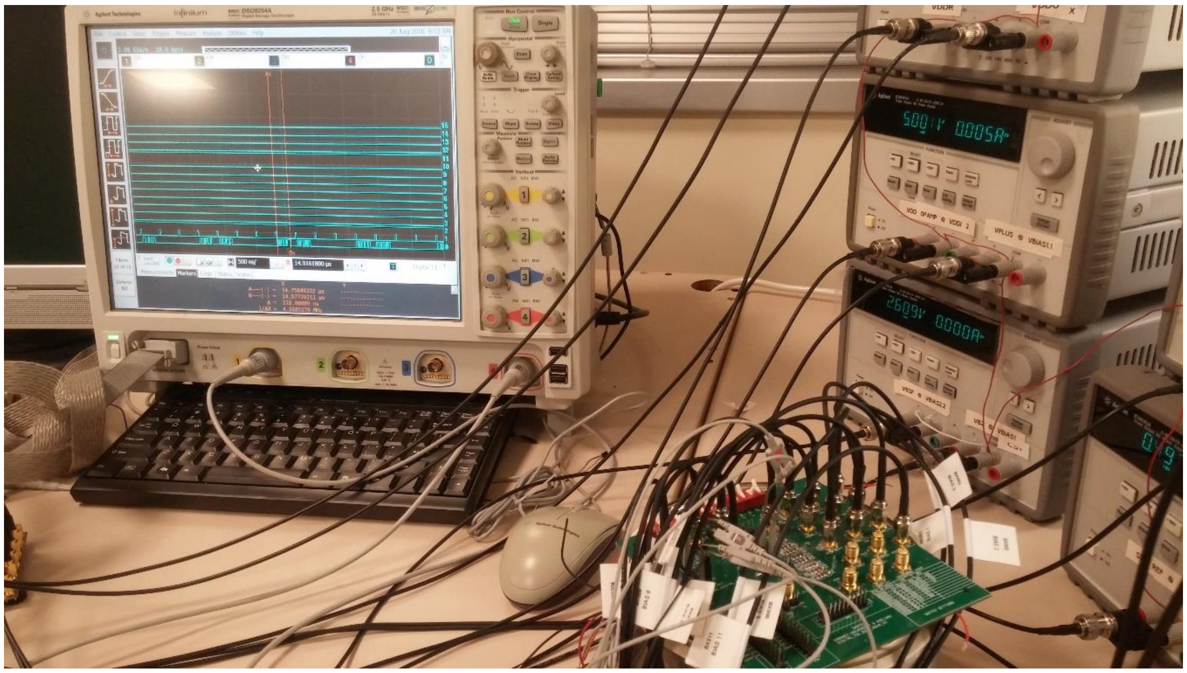


Figure 44: Photo of the measurement setup

4.2.2 Pixel Linearity

Linearity is an important parameter for the image quality. The pixel should integrate photocurrent proportional to the incoming flux. If the input photocurrent is doubled for a fixed integration time, the pixel output should also be doubled to differentiate different brightness levels successfully. Similarly, if the integration time is doubled for a fixed input photocurrent, the pixel output should also be doubled. Pixel linearity is verified by measuring the pixels with fixed input current and varying integration time. *Figure 45* shows the output with respect to varying integration time. Measurement results prove the linearity of the pixel.

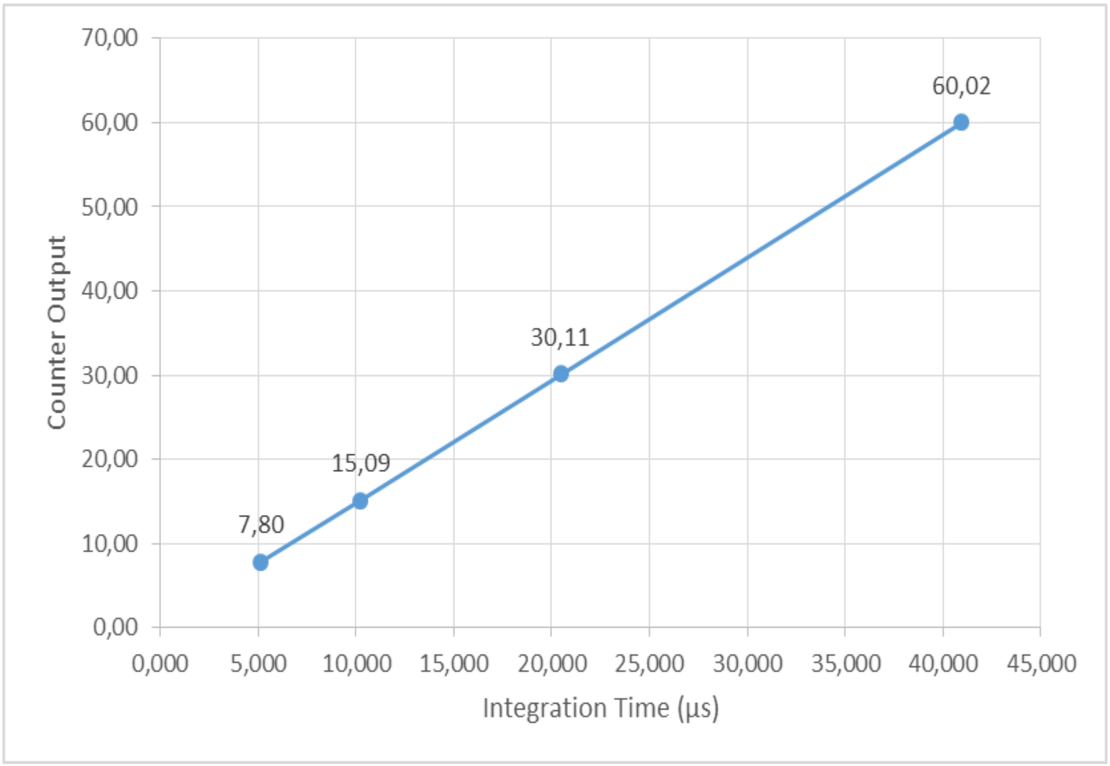


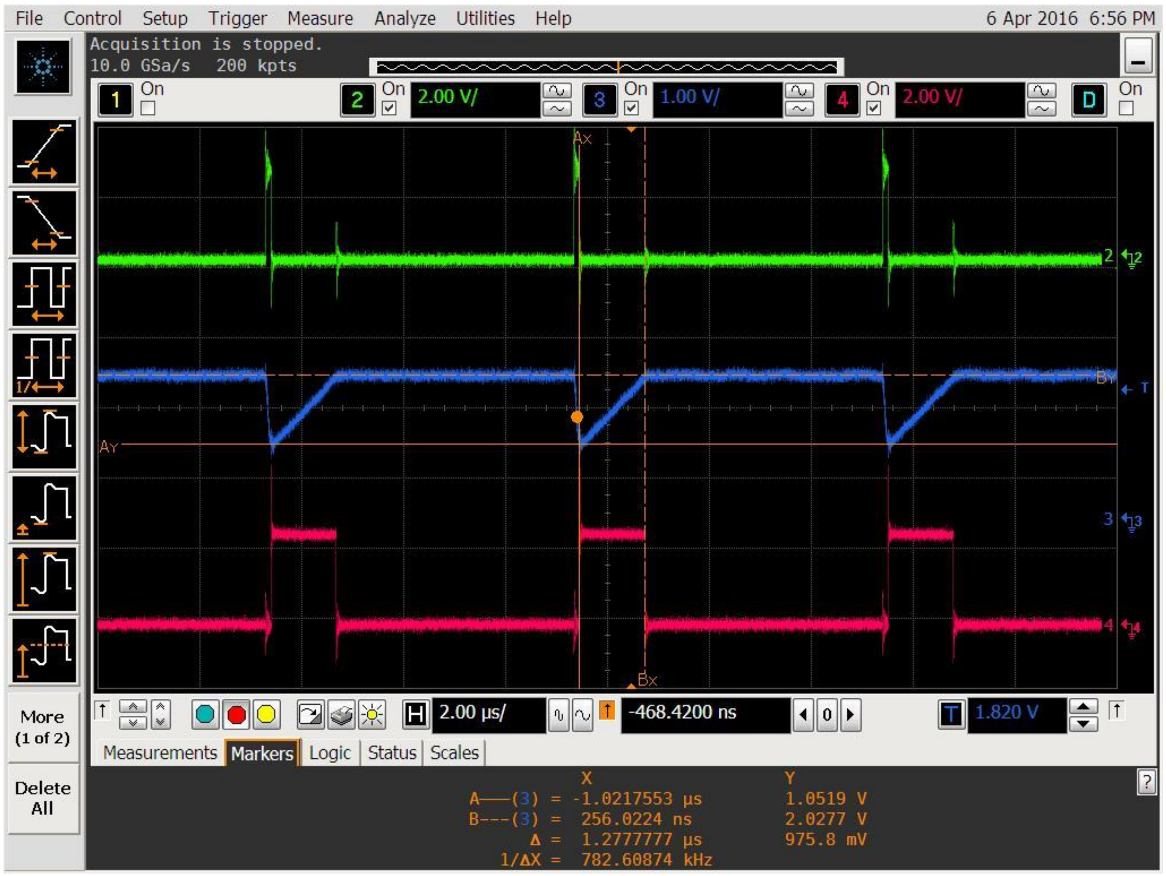
Figure 45: Pixel linearity measurement with varying integration time and fixed input current

4.2.3 Ramp Generator

There is an independent ramp generator used for the calibration of the ramp generator. This ramp generator is not connected to the 90x8 prototype array in order not to load and affect the performance of the ramp generator of the actual array. For different clock speeds and residual times, it is possible to adjust the bias voltage that is controlling the integration current of the ramp generator.

Figure 46 shows the ramp generator control signals and output. The RST RAMP signal is at the top while the RESIDUE signal is at the bottom. Ramp generator output is located in the middle of these control signals. As it is seen from the figure, the ramp generator operates as expected with changes of the control signals. The clock frequency here is 200 MHz, and the RESIDUE signal is active for 256 clock cycles. This can be understood by dividing the 1.277

µs, that is the time difference between the two markers of the oscilloscope, to 256 cycles. The sweep range starts from 1.05 V and ends at 2.02 V, which means 975 mV sweep range.



```
Acquisition is stopped.
10.0 GSa/s   200 kpts
```

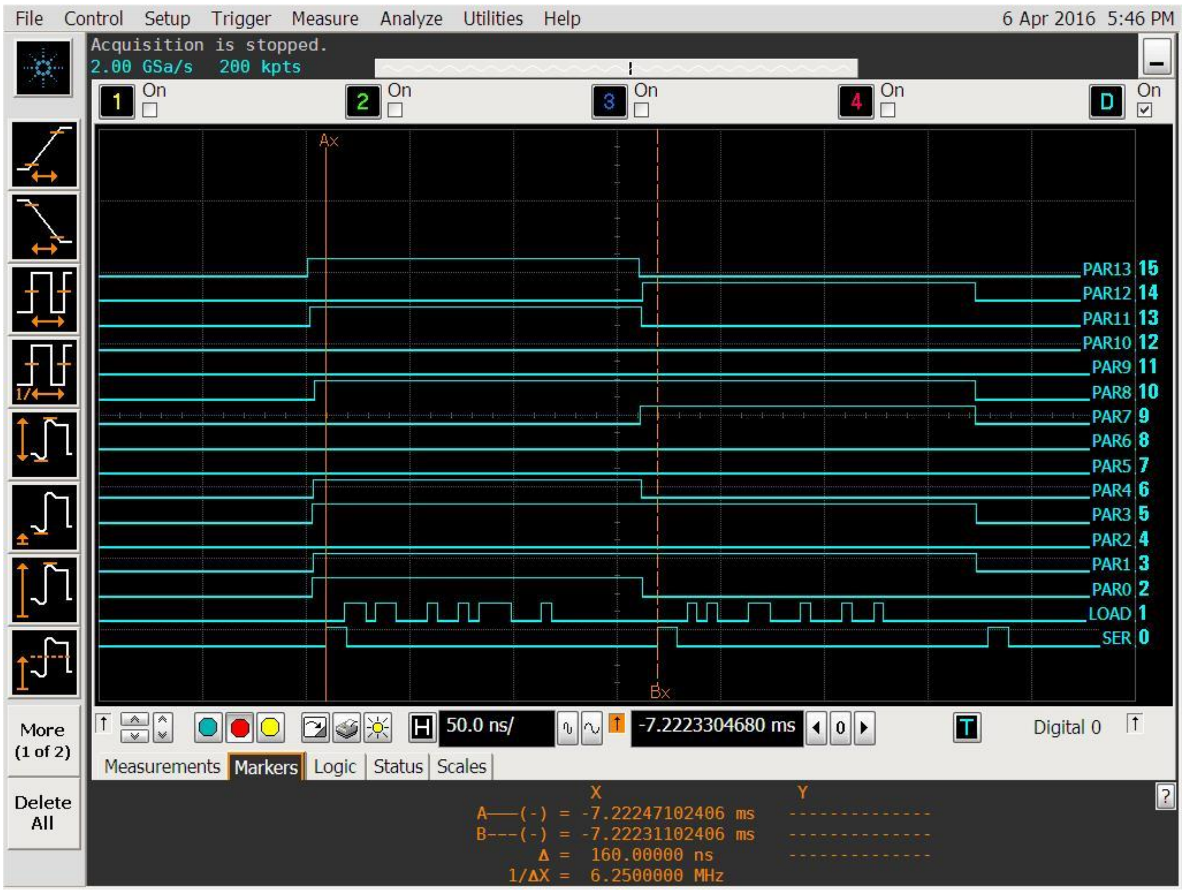
```
                  X                  Y
A——(3) =  -1.0217553 µs        1.0519 V
B---(3) =   256.0224 ns        2.0277 V
      Δ =   1.2777777 µs       975.8 mV
   1/ΔX =   782.60874 kHz
```

Figure 46: Measurement result of the ramp generator

4.2.4 Serializer

Figure 47 shows the measurement results of the serializer. 14-bit of the 22-bit output of the array are shown together with LOAD and SER signals. LOAD signal is used to capture the 22-bit counter output from the multiplexed row. SER signal is the output signal of the serializer. Since the number of digital oscilloscope probes is limited to 16 bits, just 14 bits of the multiplexed parallel input are seen. At the positive edge of the LOAD signal (marker A), parallel data is captured as the following starting from PAR0: 1-1-0-1-1-0-0-0-1-0-0-1-0-1-

1-1-0-0-0-1-0-0. When the LOAD signal is deactivated, the serializer starts to output these captured bits one by one. The same sequence can be followed by the SER output in *Figure 47* starting from the least significant bit (PAR0). This shows the serializer works as expected.



```
Acquisition is stopped.
2.00 GSa/s   200 kpts
50.0 ns/     -7.2223304680 ms

                 X                         Y
A——(-) = -7.22247102406 ms    --------------
B---(-) = -7.22231102406 ms    --------------
      Δ =  160.00000 ns       --------------
   1/ΔX =  6.2500000 MHz
```

Figure 47: Measurement result of the serializer

4.2.5 Prototype Array

The charge handling capacity, the quantization noise, the signal-to-noise ratio, the power dissipation of the proposed digital TDI ROIC are verified at cryogenic temperature (77K). As explained in the third section, the charge packet associated with a single count of the MSB counter is 175.000 e-. A maximum charge-handling capacity of 44.8 Me- is verified by saturating the MSB counter with a sufficient amount of input current and integration time.

The quantization noise of the proposed ROIC is verified by just activating the LSB counter (single slope ramp ADC) of the ROIC with limited input current and integration time. The total resolution is 16 bits with 8 bits of MSB and 8 bits of LSB.

The proposed ROIC is measured with various input currents and integration times. With fixed input current, doubling the integration time also doubled the output count. The same behavior is observed by repeating the same operation several times. Since almost all blocks are connected to different I/O pads, it is possible to measure the power dissipation of all blocks separately. *Table 9* shows the power dissipation of various blocks of the digital TDI ROIC with nominal switching activity with an input current of 10 nA and maximum switching activity with an input current of 30 nA. Since the proposed ROIC is a PFM based architecture, with increasing input current and integration time, the switching activity of the pixels and counters also increases, which results in an increased dynamic power consumption. This behavior is also verified with the measurements.

Table 9: Power dissipation of the 90x8 digital TDI ROIC for nominal and maximum switching activities at 77 K

Block	Average Current (Max)	Average Current (Nominal)	Supply Voltage (V)	Power Dissipation (Max)	Power Dissipation (Nominal)
Pixel + Ramp Generator	3.56 mA	3.02 mA	2.5	8.9 mW	7.55 mW
Control Block	1.25 mA	1.25 mA	1.8	2.25 mW	2.25 mW
Counters	2.05 mA	1.77 mA	1.8	3.69 mW	3.186 mW
Analog I/O Pads	7.5 µA	5.39 µA	3.3	24.75 µW	17.787 µW
Digital I/O Pads	0.55 mA	0.41 mA	3.3	1.815 mW	1.353 mW
Total				16.779 mW	14.356 mW

The proposed digital TDI ROIC has also been evaluated in terms of noise performance. It has been observed that with the increasing switching activity the measured SNR of the ROIC deviates from the shot noise limit. This deviation from shot noise limit is observed with both increasing input current and integration time hence can be linked with switching noise. Similar behavior related to the switching noise is also reported in some DROIC works [56] [59]. In the low illumination conditions where the collected electrons are less than 4 Me-, excellent SNR results are observed due to the significantly reduced quantization noise with the use of the single slope ramp ADC and the effect of TDI architecture. *Figure 48* shows the SNR graph for low the illumination conditions while *Figure 49* shows the SNR graph for the whole dynamic range. The minimum SNR value for the low illumination conditions is measured as 51 dB, which is a very good SNR value for the collected electrons of 280 Ke$^-$. The measured peak SNR value for the digital TDI ROIC is 72 dB. If the switching noise performance of the ROIC is improved with a better isolation of the sensitive comparator from the digital signals, better SNR values for the high illumination conditions are also possible.

Table 10 shows the performance summary of the proposed digital TDI ROIC. ROIC can operate up to 200 MHz. For an image format of 360x90, 192 Hz frame rate is achievable. If 4 of 90x8 ROICs are stitched together, it is possible to have an image resolution of 360x360. Single pixel charge handling capacity is 44.8 Me$^-$ resulting in 358 Me$^-$ for 8 TDI pixels. 198 e$^-$ quantization noise offers significantly improved SNR values for low illumination conditions, which is the main motivation factor for most TDI applications. 14.5 mW power consumption is also very important for the SWaP trend of infrared imaging systems.

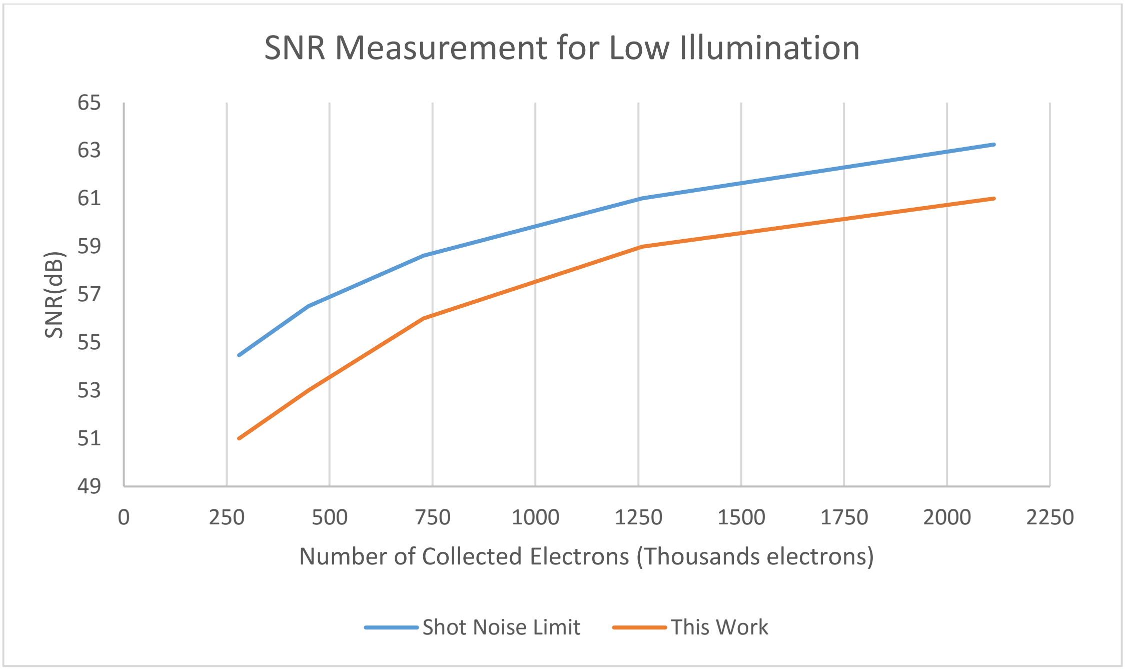


Figure 48: SNR measurement for the low illumination conditions

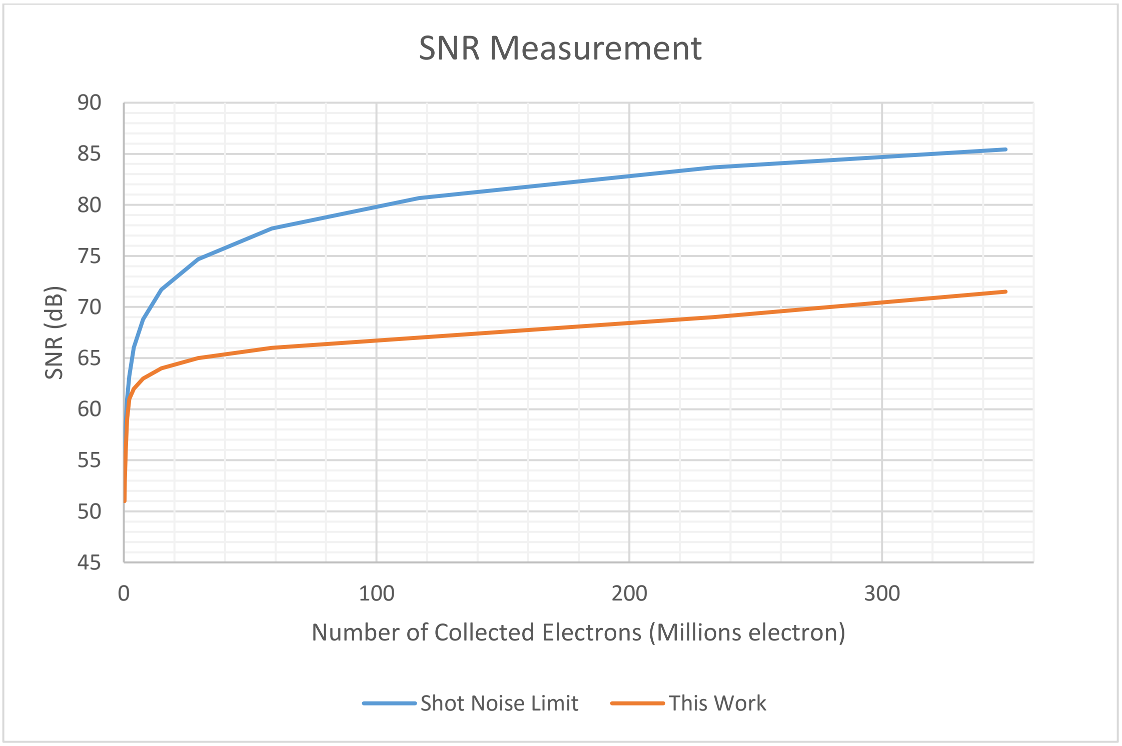


Figure 49: SNR measurement for the whole dynamic range

Table 10: Performance parameters of the proposed digital TDI ROIC

Parameter	**Value**
Array Format	90x8
Infrared Band	LWIR
Operating Temperature	77 K
Pixel Dimensions	15 µm X 15 µm
CMOS Process Node	180 nm
ROIC Type	Digital Pixel
Oversampling Rate	2
Input Current	1-50 nA
Resolution	16-bits (8-bits MSB, 8-bits LSB)
Operating Frequency	Up to 200 MHz
Frame Rate	Up to 192 Hz (for 360x90 image format) or 70 K lines/s
Power Dissipation	14.35 mW (for input current of 10 nA)
Charge Handling Capacity	44.8 Me$^-$ (for single pixel)
Quantization Noise	198 e$^-$
SNR	72 dB
Programming	Serial / Parallel programming, Bypass mode, bi-directional scanning capability, adjustable integration time, pixel deselection

5. CONCLUSION

A new digital pixel ROIC, implementing TDI method for the scanning type IRFPA arrays, is proposed and a 90x8 prototype array is fabricated in 180 nm CMOS process. The proposed DROIC is based on the PFM architecture and improved with a single slope ramp ADC. 51 dB SNR is achieved for 280.000 collected electrons with the help of 198 e^- reduced quantization noise, which improves the SNR, especially in the low illumination conditions. Measurement results show that for the high illumination conditions switching noise limits the SNR performance of digital TDI ROIC. However, it is still possible to achieve high SNR values such as 80 dB for 350 Me^- collected electrons from 8 TDI pixels. Due to the switching noise limitations, 72 dB peak SNR is measured for an input current of 10 nA. The proposed digital TDI ROIC is also a power efficient architecture compared to the analog TDI implementations. For a nominal input current of 10 nA, 14.35 mW power dissipation is measured, which is lower than any analog TDI examples.

5.1.1 Comparison of the Thesis Work

Table 11 shows the comparison of the proposed digital TDI ROIC with the analog and the digital TDI ROICs. Almost all of the digital TDI ROICs are designed for CMOS image sensor applications, which dictates the use of the column parallel ADC architecture for smaller pixel pitches. Since CMOS image sensors are exposed to the visible range of the EM spectrum, their charge handling capacity is limited to kilo electrons range. Due to the limited charge-handling capacity, CMOS image sensors have a noise specification of a couple of electrons. Since the conventional 4T pixel architecture is used in CMOS image sensors, they do not need a large pixel pitch. Since they do not need large pixels, they also do not need oversampling to improve the spatial resolution. Due to these different specifications related to the CMOS image sensor applications, it is not fair to compare the performance parameters of these ROICs with the TDI ROICs designed for LWIR applications. However, there is only one example of the digital TDI ROIC for LWIR applications. Because of the single example of TDI ROIC for LWIR applications, TDI ROIC examples for CMOS image sensors are also put on the table. All the digital TDI ROICs for CMOS image sensors exhibit low power consumption per pixel. For example, 1024x128 array of [51] consumes 290 mW power, which means 2.21 μW per pixel. This digital TDI ROIC uses a cyclic ADC together with a digital accumulator. One of the reasons that this circuit consumes less power is the speed of the ROIC, which is 3875 lines/s.

The only example of the digital TDI ROIC for the LWIR applications is the MIT Lincoln Lab's DROIC. This DROIC uses the conventional PFM architecture in order to have a large charge handling capacity, which is 260 Me$^-$ for the TDI application. In fact, this DROIC is not designed specifically for the TDI applications but is shown to operate for the TDI operation successfully. Since 16-bit counter and orthogonal transfer registers to implement the TDI functionality are put in the pixel, this DROIC needs at least 30 μm pixel pitch although it uses 90 nm CMOS process. Considering the SWaP trend and small pixel pitches of the detector vendors, this DROIC with 30 μm pixel pitch cannot be used for a long time. To overcome this problem even more advanced CMOS process nodes such as 28 nm is needed for this type of DROIC architectures. Another disadvantage of this DROIC is the high quantization noise due to the large charge packet size. The main advantage of this DROIC is the very low power consumption per pixel and very high data rate compared to the other digital TDI ROIC examples. It is also possible to operate these ROICs with high frame rates.

There are many examples of the analog TDI ROICs in the market and in the literature. Almost all analog ROICs except the one in [60] occupy large pixel area and consume a large amount of power due to the reasons mentioned in Section 2.2. They are also limited in the charge handling capacity. The 128-stage analog accumulator TDI ROIC [60] has the same pixel pitch with this thesis work and lower power consumption per pixel. The first reason why this ROIC consumes less power than our thesis work is the operation speed of the ROIC, which is approximately 15 times slower than this thesis work. Another important factor for the lower power consumption is that the analog accumulator TDI ROIC is designed for CMOS image sensors, which have a very limited charge handling capacity. In other words, it eliminates the use of the in-pixel comparator that is used in this thesis work and consumes most of the ROIC power according to Table 9, and just operates with the LSB part of this thesis work. The analog accumulator ROIC also does not use oversampling which also saves a considerable amount of power. For example, if this thesis work does not use the oversampling to improve the spatial resolution, the counter, and the control circuit blocks will consume half the amount of the current power consumption.

Table 11: Comparison of the thesis work with some analog and state of the art digital TDI ROICs

	ROIC Type	**Detector Type**	**Tech. Node**	**Over-samp-ling**	**Speed**	**Format**	**Pixel Pitch (µm)**	**Charge hand-ling capa.**	**Power dis. per pixel (µW)**	**Noise**
SOFRADIR [61]	Analog	LWIR	N/A	Yes / 3	50 Hz	480x6	50x25	12.5	N/A	N/A
SOFRADIR [62]	Analog	LWIR	N/A	Yes / 3	50 Hz	288x4	43x28	16	N/A	N/A
SCD [63]	Analog	LWIR	N/A	N/A	50 Hz	480x6	N/A	N/A	156.25	N/A
Key Laboratory [64]	Analog	N/A	500 nm	Yes / 3	50 Hz	288x4	56x43	12.5	69.44	2000e$^-$
Chung-Shan Institute [16]	Analog	LWIR	350 nm	No	50 Hz	32x4	30x30	6.75	781.25	N/A
Japan Defense Agency [20]	Analog	LWIR	800 nm	N/A	30 Hz	480x8	26x26	N/A	N/A	1300e$^-$
Institute of Microdevices [65]	Analog	LWIR	600 nm	No	50 Hz	576x6	56x43	17 Me$^-$	28.93	1300e$^-$
Tianjin University [60]	Analog	CMOS	180 nm	No	3875 lines/s	1024x128	15x15	NA	3.81	N/A
MIT Lincoln Lab	Digital	LWIR	90 nm	No	155 Hz	256x256	30x30	260 Me$^-$	1.02	N/A
CMOSIS	Digital	CMOS	180 nm	No	340 fps	2048x1088	5.5x5.5	18 Ke$^-$	0.26	13e$^-$
Tianjin University [51]	Digital	CMOS	180 nm	No	3875 lines/s	1024x128	15x15	N/A	2.21	N/A
This work	Digital	LWIR	180 nm	Yes / 2	70 K lines/s	90x8	15x15	358 Me$^-$	17.90	1260

As a result, the digital TDI ROIC, that is proposed in this thesis work, offers smaller pixel pitch, higher charge handling capacity, lower power consumption and comparable noise performance compared to the analog TDI ROICs. With a careful design and layout by paying attention to the isolation of the pixel comparators from the digital circuitry, the digital TDI ROIC also exhibits the potential for better noise performance. The proposed digital TDI ROIC implements the PFM architecture to scanning array systems and improves the quantization noise of the PFM architecture with a single slope ramp ADC. The digital TDI ROIC does not require a large integration capacitor in-pixel like in analog TDI ROICs, hence enables smaller pixel pitch and higher charge handling capacity, which is in line with the SWaP trend. Compared to the digital TDI ROIC examples using the column parallel ADC architecture, the implemented digital TDI ROIC in this thesis has better signal integrity due to the digital signal chain. Considering the different performance parameters such as oversampling and charge handling capacity, the digital TDI ROIC implemented in this thesis work also exhibits similar power consumption with an improved quantization noise, hence improved noise and SNR performance. The proposed digital TDI ROIC also has larger charge handling capacity compared to the column parallel ADC architectures. Compared to the single example of digital TDI ROIC for LWIR applications, the proposed digital TDI ROIC has smaller pixel pitch and better quantization noise. Although the digital TDI ROIC in [53] has a better power consumption performance compared to this thesis work, it requires large pixel pitch as opposed to the small pixel pitch trend of the industry. It also requires advanced process node and has a large quantization noise.

5.1.2 Suggestions for Further Improvements

The first performance parameter of the thesis work to be improved is the pixel pitch. Studies carried out during the thesis work has shown that it is possible to have a pixel pitch of 10 µm with this digital TDI ROIC design. If the pixel pitch is selected as 10 µm, there is no need for the oversampling. Removing the oversampling property from the ROIC will eliminate the use of extra 7 MSB and LSB counters. Reduction in the number of counters will also result in less number of control signals and less complex control circuit. All these improvements will be reflected as a power reduction and area reduction for the next generation digital TDI ROIC.

Another possible improvement for the current version of the digital TDI ROIC is to use an improved DFF architecture regarding area constraints. In this thesis work, s standard DFF is used as the building block of the ripple carry counter. There are more area efficient dynamic flip-flops in the literature that can be potentially used in the next generation digital TDI ROIC. Since the primary objective of this thesis is to prove the benefits of using digital ROIC architectures for the TDI applications, the area constraint was the secondary issue concerning the DFFs. Since the use of dynamic flip-flops brings extra complexity to the system, it is not desirable to take any risk for the first version of the digital TDI ROIC.

To further reduce the power consumption of the next generation digital TDI ROIC, the pixel comparator can be improved with a more power efficient one. Table 9 shows that the pixel comparator is the most power hungry block of the digital ROIC. In addition to an improved pixel comparator, instead of an op-amp based integrator for the ramp generator, a DAC based ramp generator will be better for the power and noise performance of the digital TDI ROIC. The DAC based ramp generator is not preferred due to the similar reasons with the dynamic flip-flops that can be utilized for the ripple carry counters. Finally, the use of a more advanced CMOS process node such as 90 nm, will also reduce the power consumption of the next generation ROIC. An advanced process node will also make it easier to fit into 10 μm X 10 μm pixel area with a fabrication cost trade-off.

6. REFERENCES

[1] W. Herschel, "Experiments on the Refrangibility of the Invisible Rays of the Sun," *Philosophical Transactions of the Royal Society of London,* vol. 90, pp. 284-292, 1800.

[2] N. K. Dhar, R. Dat and A. K. Sood, "Advances in Infrared Detector Array Technology," in *Optoelectronics - Advanced Materials and Devices*, InTech, 2013, pp. 149-190.

[3] A. Rogalski, "Third-generation infrared photon detectors," in *SPIE*, 2003.

[4] A. Rogalski, "History of infrared detectors," *Opto-electronics Review,* vol. 20, no. 3, pp. 279-308, 2012.

[5] C. C. Kim and R. Meyer, "Comparison of IRST systems by SNR," in *SPIE 9071 Infrared Imaging Systems: Design, Analysis, Modelingi and Testing XXV*, Baltimore, 2014.

[6] G. Barani, M. Olivieri, C. Luison, A. Rossi, M. Diani and N. Acito, "Development of a panaromic third generation IRST: initial study and experimental work," in *SPIE 8704 Infrared Technology and Applications XXXIX*, Baltimore, 2013.

[7] T. E. White and A. R. Leary, "Digital IR imaging capability for medical applications," in *SPIE 3712 Battlefield Biomedical Technologies*, Orlando, 1999.

[8] C.-C. Hsieh, C.-Y. Wu, F.-W. Jih and T.-P. Sun, "Focal-Plane-Arrays and CMOS Readout Techniques of Infrared Imaging Systems," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 7, no. 4, pp. 594-605, AUGUST 1997.

[9] D. Rock, J. Redus, L. Marr, M. Gohlke and D. Hartnett, "Falcon Eye Forward-Looking Infrared (FLIR) System," in *SPIE 0782 Infrared Sensors and Sensor Fusion*, Orlando, 1987.

[10] L. Dan, T. Ju, L. Wengao, C. Zhongjian, Z. Baoying and J. Lijiu, "Low Power Design of Column Readout Stage for 320x288 Snaphot Infrared ROIC," in *IEEE Conference on Electron Devices and Solid-State Circuits*, 2005.

[11] K. C. Liddiard, "Novel concepts for low-cost IR security sensors," in *SPIE Infrared Technology and Applications XXXI*, Orlando, 2005.

[12] U. Hoffmann, G. Hofmann, D. Wassilew, N. Heb and M. Zimmerhackl, "High temperature IR-imager with wide dynamic range for industrial process control," in *SPIE 6939 Thermosense XXX*, Orlando, 2008.

[13] P. Tribolet and G. Destefanis, "Third generation and multi-color IRFPA developments: a unique approach based on DEFIR," in *SPIE*, Orlando, 2005.

[14] A. Ezerskaia, S. F. Pereira, H. P. Urbach and B. Varghese, "Inter- and intra-individual differences in skin hydraion and surface lipids measured with mid-infarared spectroscopy," in *SPIE Optical Diagnostics and Sensing XVI: Toward Point-of-Care Diagnostics*, 2016.

[15] "Applications," Satellite Imaging Corporation, [Online]. Available: http://www.satimagingcorp.com/applications/. [Accessed 26 August 2016].

[16] "PathFindIR II Driver Vision Enhancement System," [Online]. Available: cvs.flir.com/pathfinderII-datasheet. [Accessed 28 August 2016].

[17] "FLIR One," [Online]. Available: www.flir.eu/flirone/ios-android. [Accessed 28 August 2016].

[18] G. C. Holst, Testing and Evaluation of Infrared Imaging Systems, Florida: JCD Publishing and SPIE Press, 2008.

[19] H. Kaplan, Practical Applications of Infrared Thermal Sensing and Imaging Equipment, SPIE, 2007.

[20] W. Radford, E. Patten, D. King, G. Pierce, J. Vodicka, P. Goetz, G. Venzor, E. Smith, R. Graham, S. Johnson, J. Roth, B. Nosho and J. Jensen, "Third Generation FPA Development Status at Raytheon Vision Systems," in *SPIE Infrared Technology and Applications XXXI*, Bellingham, 2005.

[21] I. Hirsh, L. Shkedy, D. Chen, N. Fishler, Y. Hagbi, A. Koifman, Y. Openhaim, I. Vaserman, M. Singer and I. Shtrichman, "Hybrid Dual-Color MWIR Detector for Airborne Missile Warning Systems," in *SPIE Infrared Technology and Applications XXXVII*, Baltimore, 2012.

[22] R. Breiter, W. Cabanski, K. Mauk, W. Rode, J. Ziegler, H. Schneider and M. Walther, "Multicolor and Dual-Band IR Camera for Missile Warning and automatic Target Recognition," in *SPIE Targets and Backgrounds VIII: Characterization and Representation*, 2002.

[23] M. Vuillermet and P. Chorier, "Latest Sofradir technology developments usable for space applications," in *SPIE Sensors, Systems, and Next-Generation Satellites X*, 2006.

[24] C. Fink, "New technologies for HWIL testing of WFOV, large-format FPA sensor systems," in *SPIE Infrared Imaging Systems: Design, Analysis, Modeling, and Testing XXVII*, Baltimore, 2016.

[25] S. D. Gunapala, S. V. Bandara, J. K. Liu, J. M. Mumolo, D. Z. Ting, C. J. Hill, J. Nguyen, B. Simolon, J. Woolaway, S. C. Wang, W. Li, P. D. Le Van and M. Z. Tidrow, "Demonstration of Megapixel Dual-Band QWIP Focal Plane Array," *IEEE Journal of Quantum Electronics*, vol. 46, no. 2, pp. 285-293, 2010.

[26] E. Dereniak and J. Hubbs, SC152 Infrared Focal Plane Arrays Course Notes, SPIE, 2010.

[27] J. D. Vincent, S. E. Hodges, J. Vampola, M. Stegall and G. Pierce, Fundamentals of Infrared and Visible Detector Operation and Testing, Wiley, 2016.

[28] J. Vampola, "Readout Electronics for Infrared Sensors," in *The Infrared and Electro-Optical Systems Handbook*, vol. 3, SPIE, pp. 285-342.

[29] K. Veeder, Analog-to-Digital Converters for Digital Readout Integrated Circuits, SPIE, 2012.

[30] J. E. Rushton, K. D. Stefanov, A. D. Holland, J. Endicott, F. Mayer and F. Barbier, "A CMOS TDI image sensor for Earth observation," in *SPIE Nanophotonics and Macrophotonics for Space Environments IX*, 2015.

[31] N. Kai-ming, Y. Su-ying, X. Jiang-tao and G. Jing, "Modeling and Simulation of TDI CMOS Image Sensors," in *SPIE International Symposium on Photoelectronic Detection and Imaging: Infrared Imaging and Applications*, 2013.

[32] A. Eckardt, R. Reulke, M. Jung and K. Sengebusch, "CMOS-TDI detector technology for reconnaissance application," in *SPIE Electro-Optical and Infrared Systems: Technology and Applications XI*, 2014.

[33] Lohmüller, Bertram;, "TDI Time Delay Integration Cameras," Hamamatsu, 2008. [Online]. Available: http://spectronet.de/story_docs/vortraege_2008/081104_vision_2008/081106_1000_lohmueller_hamamatsu.pdf. [Accessed 25 March 2016].

[34] B. J. Chen, "Pipelined Amplifier Time Delay Integration". US Patent US 7532242 B1, 12 May 2009.

[35] F.-K. Tsai and H.-Y. Huang, "A Time-Delay-Integration CMOS Readout Circuit for IR Scanning," in *9th International Conference on Electronics, Circuits and Systems*, 2002.

[36] R. Lineback, "Seven Opto-Sensor-Discrete Products Achieved Record Sales in 2012," IC Insights, 2013.

[37] O. Ceylan, *MSc Thesis Realization of Readout Integrated Circuit (ROIC) for an Array of 72x4, P-on-N type HgCdTe Long Wave Infrared Detectors,* Sabancı University, 2008.

[38] H. Kayahan, M. Yazici, O. Ceylan, M. B. Baran and Y. Gurbuz, "Design of ROIC Based on Switched Capacitor TDI for MCT LWIR Focal Plane Arrays," in *SPIE Infrared Technology and Applications*, 2011.

[39] M. Kobayashi, H. Wada, T. Okamura, J. Kudo, K. Tanikawa, S. Hikida, Y. Miyamoto, S. Miyazaki and Y. Yoshida, "480X8 hybrid HgCdTe infrared focal plane arrays for high-definition television format," *Optical Engineering*, vol. 41, no. 8, pp. 1876-1885, 2002.

[40] E. Mottin and P. Pantigny, "Device for Reading Detector Arrays with TDI Effect". US Patent 5828408, 27 October 1998.

[41] W.-L. Wang, S. Lin, C.-P. Lin and F.-K. Hsiao, "Time Delay Integration Based MOS Photoelectric Pixel Sensing Circuit". US Patent US2011/0019044 A1, 27 January 2011.

[42] M. F. Audier, V. Besnard and G. Rigaux, "Circuit For the Reading of Linear Arrays of Photodetectors". US Patent 59998777, 7 December 1999.

[43] K.-W. Cheng, C. Yin, C.-C. Hsieh, W.-H. Chang, H.-H. Tsai and C.-F. Chiu, "Time-Delay Integration Readout with Adjacent Pixel Signal Transfer for CMOS Image Sensor," in *International Symposium on VLSI Design, Automation, and Test*, 2012.

[44] K. Faramarzpour and M. E. Sonder, "CMOS Time Delay and Integration Image Sensor". US Patent 8975570 B2, 10 March 2015.

[45] P. Pantigny, "Device for Detecting a Photonic Flux with Self-adaptive Scanning". US Patent 7358996, 15 April 2008.

[46] F. Serra-Graells, B. Misischi, E. Casanueva, C. Mendez and L. Teres, "Low-Power and Compact CMOS APS Circuits for Hybrid Cryogenic Infrared Fast Imaging," *IEEE Transactions on Circuits and Systems-II:Express Briefs*, vol. 54, no. 12, pp. 1052-1056, 2007.

[47] G. Lepage, J. Bogaerts and G. Meynants, "Time-Delay-Integration Architectures in CMOS Image Sensors," *IEEE Transactions on Electron Devices,* vol. 56, no. 11, pp. 2524-2533, 2009.

[48] J. Bogaerts and S. K. Waver, "Analog-to-digital Conversion in Pixel Arrays". US Patent 7880662 B2, 1 February 2011.

[49] Y. Muramatsu, N. Fukushima, Y. Nitta and Y. Yasui, "Method and Apparatus for AD conversion, Semiconductor Device for Detecting Distribution of Physical Quantity, and Electronic Apparatus". US Patent 7532148 B2, 12 May 2009.

[50] X. Wang, J. Bogaerts, G. Vanhorebeek, K. Ruythoren, B. Ceulemans, G. Lepage, P. Willems and G. Meynants, "A 2.2M CMOS Image Sensor for High Speed Machine Vision Applications," in *SPIE-IS&T Sensors, Cameras, and Systems for Industrial/Scientific Applications XI*, 2010.

[51] K. Nie, J. Xu and Z. Gao, "A 128-Stage CMOS TDI Image Sensor With On-Chip Digital Accumulator," *IEEE Sensors Journal,* vol. 16, no. 5, pp. 1319-1324, 2016.

[52] Y. Katzir, I. Gur-Arie and Y. Malinovich, "High-sensitivity Optical Scanning Using Memory Integration". US Patent 7129509 B2, 31 October 2006.

[53] B. Tyrrell, K. Anderson, J. Baker, R. Berger, M. Brown, C. Colonero, J. Costa, B. Holford, M. Kelly, E. Ringdahl, K. Schultz and J. Wey, "Time Delay Integration and In-Pixel Spatiotemporal Filtering Using a Nanoscale Digital CMOS Focal Plane Readout," *IEEE Transactions on Electron Devices,* vol. 56, no. 11, pp. 2516-2523, 2009.

[54] M. Kelly, B. Tyrrell, C. Colonero, R. Berger, K. Schultz, J. Wey, D. Mooney and C. Lawrence, "Focal Plane Array Processing Method and Apparatus". US Patent 2014/0197303 A1, 17 July 2014.

[55] T. K. Veeder, "Time-frequency fusion Digital Pixel Sensor". US Patent 8158923 B2, 17 April 2012.

[56] H. Kayahan, M. Yazici, Ö. Ceylan and Y. Gurbuz, "A new digital readout integrated circuit (DROIC) with pixel parallel A/D conversion and reduced quantization noise," *Infrared Physics & Technology,* vol. 63, pp. 125-132, 2014.

[57] M. Bazes, "Two Novel Fully Complementary Self-Biased CMOS Differential Amplifiers," *IEEE Journal of Solid-State Circuits,* vol. 26, no. 2, pp. 165-168, 1991.

[58] H. Kayahan, O. Ceylan, M. Yazici, S. Zihir and Y. Gurbuz, "Wide Range, Process and Temperature Compansated Voltage Controlled Current Source," *IEEE Transactions on Circuits and Systems I: Regular Papers,* vol. 60, pp. 1345-1353, 2013.

[59] S. Bisotto, E. d. Borniol, L. Mollard, F. Guellec, A. Peizerat and M. Tchagaspanian, "A 25 um pitch LWIR staring focal plane array with pixel-level 15-bit ADC ROIC achieving 2mK NETD," in *SPIE Electro-Optical Infrared Systems, Technology, Applications VII,* 2010.

[60] N. Kaiming, S. Yao, J. Xu, J. Gao and Y. Xia, "A 128-Stage Analog Accumulator for CMOS TDI Image Sensor," *IEEE Transactions on Circuits and Systems-I: Regular Papers,* vol. 61, no. 7, pp. 1952-1961, 2014.

[61] SOFRADIR, [Online]. Available: http://www.sofradir.com/wp-content/ uploads/2013/09/ Fiche_mercury-lw_web.pdf. [Accessed 20 June 2016].

[62] SOFRADIR, [Online]. Available: http://www.sofradir.com/ wp-content/ uploads /2013/09/ Fiche_Pluton-lw_web.pdf. [Accessed 20 June 2016].

[63] "TDI 480x6," SCD USA, [Online]. Available: http://www.scdusa-ir.com/tdi-480x6/. [Accessed 20 June 2016].

[64] Z. Chen, W. Lu, J. Tang, Y. Zhang, C. Junmin and L. Ji, "A CMOS TDI readout circuit for infrared focal plane array," in *International Conference on Solid-State and Integrated-Circuit Technology (ICSICT)*, 2008.

[65] F. F. Sizov, Y. P. Derkach, S. V. Korinets and V. P. Reva, "576x6 ROIC for MCT LWIR Arrays," in *SPIE Semiconductor Photodetectors II*, Bellingham, 2005.

7. APPENDIX

7.1 Verilog Codes

7.1.1 Control Circuit

```
`timescale 1ns/1ps
module dec2to4 (enable, dec_in, dec_out);

input enable;
input [1:0] dec_in;
output [3:0] dec_out;

reg [3:0] dec_out;

always@(enable or dec_in)

        case ( {enable,dec_in} )

        3'b 100: dec_out = 4'b 0001;
        3'b 101: dec_out = 4'b 0010;
        3'b 110: dec_out = 4'b 0100;
        default: dec_out = 4'b 0000;

        endcase
endmodule


`timescale 1ns/1ps
module dec5to32 (enable, in, out);

input enable;
input [4:0] in;
output [31:0] out;

reg [31:0] out;

always@(in or enable)

        case ( {enable,in} )

        6'b100000: out = 32'b 0000_0000_0000_0000_0000_0000_0000_0001;
        6'b100001: out = 32'b 0000_0000_0000_0000_0000_0000_0000_0010;
        6'b100010: out = 32'b 0000_0000_0000_0000_0000_0000_0000_0100;
        6'b100011: out = 32'b 0000_0000_0000_0000_0000_0000_0000_1000;
        6'b100100: out = 32'b 0000_0000_0000_0000_0000_0000_0001_0000;
        6'b100101: out = 32'b 0000_0000_0000_0000_0000_0000_0010_0000;
        6'b100110: out = 32'b 0000_0000_0000_0000_0000_0000_0100_0000;
        6'b100111: out = 32'b 0000_0000_0000_0000_0000_0000_1000_0000;
        6'b101000: out = 32'b 0000_0000_0000_0000_0000_0001_0000_0000;
        6'b101001: out = 32'b 0000_0000_0000_0000_0000_0010_0000_0000;
        6'b101010: out = 32'b 0000_0000_0000_0000_0000_0100_0000_0000;
```

```
		6'b101011: out = 32'b 0000_0000_0000_0000_0000_1000_0000_0000;
		6'b101100: out = 32'b 0000_0000_0000_0000_0001_0000_0000_0000;
		6'b101101: out = 32'b 0000_0000_0000_0000_0010_0000_0000_0000;
		6'b101110: out = 32'b 0000_0000_0000_0000_0100_0000_0000_0000;
		6'b101111: out = 32'b 0000_0000_0000_0000_1000_0000_0000_0000;
		6'b110000: out = 32'b 0000_0000_0000_0001_0000_0000_0000_0000;
		6'b110001: out = 32'b 0000_0000_0000_0010_0000_0000_0000_0000;
		6'b110010: out = 32'b 0000_0000_0000_0100_0000_0000_0000_0000;
		6'b110011: out = 32'b 0000_0000_0000_1000_0000_0000_0000_0000;
		6'b110100: out = 32'b 0000_0000_0001_0000_0000_0000_0000_0000;
		6'b110101: out = 32'b 0000_0000_0010_0000_0000_0000_0000_0000;
		6'b110110: out = 32'b 0000_0000_0100_0000_0000_0000_0000_0000;
		6'b110111: out = 32'b 0000_0000_1000_0000_0000_0000_0000_0000;
		6'b111000: out = 32'b 0000_0001_0000_0000_0000_0000_0000_0000;
		6'b111001: out = 32'b 0000_0010_0000_0000_0000_0000_0000_0000;
		6'b111010: out = 32'b 0000_0100_0000_0000_0000_0000_0000_0000;
		6'b111011: out = 32'b 0000_1000_0000_0000_0000_0000_0000_0000;
		6'b111100: out = 32'b 0001_0000_0000_0000_0000_0000_0000_0000;
		6'b111101: out = 32'b 0010_0000_0000_0000_0000_0000_0000_0000;
		6'b111110: out = 32'b 0100_0000_0000_0000_0000_0000_0000_0000;
		6'b111111: out = 32'b 1000_0000_0000_0000_0000_0000_0000_0000;
		default: out = 32'b 0000_0000_0000_0000_0000_0000_0000_0000;

		endcase
endmodule

`timescale 1ns/1ps
module SC (clk, INT, rst, RST_counter, rst_SC, rst_SC_counter, rst_SC_1_counter, SC_1_RST,
SC_1_enable, SC_1, SC_1_counter,
		SC_2_RST, SC_2_enable, SC_2, SC_3_RST, SC_3_enable, SC_3,
		SC_4_RST, SC_4_enable, SC_4, SC_5_RST, SC_5_enable, SC_5,
		SC_6_RST, SC_6_enable, SC_6, SC_7_RST, SC_7_enable, SC_7,
		SC_8_RST, SC_8_enable, SC_8, SC_9_RST, SC_9_enable, SC_9,
		SC_10_RST, SC_10_enable, SC_10, SC_11_RST, SC_11_enable, SC_11,
		SC_12_RST, SC_12_enable, SC_12, SC_13_RST, SC_13_enable, SC_13,
		SC_14_RST, SC_14_enable, SC_14, SC_15_RST, SC_15_enable, SC_15, SC_16_RST,
SC_16_enable, SC_16, SC_read, SC_read_final);

input clk, INT, rst, RST_counter;

output rst_SC, rst_SC_counter, rst_SC_1_counter;
output [1:0] SC_1_counter;
output [16:1] SC_read;
output [16:1] SC_read_final;

output SC_1_enable;
output SC_2_enable;
output SC_3_enable;
output SC_4_enable;
output SC_5_enable;
output SC_6_enable;
output SC_7_enable;
output SC_8_enable;
output SC_9_enable;
output SC_10_enable;
output SC_11_enable;
output SC_12_enable;
output SC_13_enable;
output SC_14_enable;
output SC_15_enable;
output SC_16_enable;

output SC_1_RST;
output SC_2_RST;
output SC_3_RST;
output SC_4_RST;
output SC_5_RST;
output SC_6_RST;
output SC_7_RST;
output SC_8_RST;
output SC_9_RST;
output SC_10_RST;
output SC_11_RST;
output SC_12_RST;
output SC_13_RST;
output SC_14_RST;
output SC_15_RST;
output SC_16_RST;

output [2:0] SC_1;
output [2:0] SC_2;
output [2:0] SC_3;
output [2:0] SC_4;
output [2:0] SC_5;
output [2:0] SC_6;
output [2:0] SC_7;
output [2:0] SC_8;
output [2:0] SC_9;
output [2:0] SC_10;
output [2:0] SC_11;
output [2:0] SC_12;
output [2:0] SC_13;
output [2:0] SC_14;
output [2:0] SC_15;
output [2:0] SC_16;


reg rst_SC;
reg rst_SC_counter;
reg rst_SC_1_counter;

reg [1:0] SC_1_counter;
reg [16:1] SC_read;

reg SC_1_enable;
reg SC_2_enable;
reg SC_3_enable;
reg SC_4_enable;
reg SC_5_enable;
reg SC_6_enable;
reg SC_7_enable;
reg SC_8_enable;
reg SC_9_enable;
reg SC_10_enable;
reg SC_11_enable;
reg SC_12_enable;
reg SC_13_enable;
reg SC_14_enable;
reg SC_15_enable;
reg SC_16_enable;
reg SC_17_enable;

wire [16:1] SC_read_final;

wire SC_1_RST;
wire SC_2_RST;
wire SC_3_RST;
wire SC_4_RST;
wire SC_5_RST;
wire SC_6_RST;
wire SC_7_RST;
wire SC_8_RST;
wire SC_9_RST;
wire SC_10_RST;
wire SC_11_RST;
wire SC_12_RST;
wire SC_13_RST;
wire SC_14_RST;
wire SC_15_RST;
wire SC_16_RST;

reg [2:0] SC_1;
reg [2:0] SC_2;
reg [2:0] SC_3;
reg [2:0] SC_4;
reg [2:0] SC_5;
reg [2:0] SC_6;
reg [2:0] SC_7;
reg [2:0] SC_8;
reg [2:0] SC_9;
reg [2:0] SC_10;
reg [2:0] SC_11;
reg [2:0] SC_12;
reg [2:0] SC_13;
reg [2:0] SC_14;
reg [2:0] SC_15;
reg [2:0] SC_16;


always@(posedge clk)
begin
        if(rst)
                rst_SC <= 1;
        else
                rst_SC <= 0;
end

always@(posedge clk)
```

```
begin
	if(rst_SC)
		rst_SC_counter <= 1;
	else
		rst_SC_counter <= 0;
end

always@(posedge clk)
begin
	if(rst_SC_counter)
		rst_SC_1_counter <= 1;
	else
		rst_SC_1_counter <= 0;
end


always@(posedge INT or posedge rst_SC_1_counter)
begin
	if(rst_SC_1_counter)
		begin
		SC_1_counter <= 0;
		end

	else if(SC_1_counter == 1)
		begin
		SC_1_counter <= 0;
		end

	else
		begin
		SC_1_counter <= SC_1_counter + 1;
		end
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
	if(rst_SC_1_counter)
		SC_1_enable <= 0;

	else if (SC_1_counter == 2'b 00)
		SC_1_enable <= 1;
	else
		SC_1_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
	if(rst_SC_1_counter)
		SC_1 <= 0;
	else if(SC_2_enable)
		SC_1 <= SC_1 + 1;
	else
		SC_1 <= SC_1;
end

always@(posedge INT or posedge rst_SC_1_counter)
```

```
begin
        if(rst_SC_1_counter)
                SC_read[1] <= 0;
        else if(SC_1 == 3'b111 && SC_1_counter == 2'b01)
                SC_read[1] <= 1;
        else
                SC_read[1] <= 0;
end


always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_2 <= 0;
        else if(SC_3_enable)
                SC_2 <= SC_2 + 1;
        else
                SC_2 <= SC_2;
end


always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_read[2] <= 0;
        else if(SC_read[1])
                SC_read[2] <= 1;
        else
                SC_read[2] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_2_enable <= 0;
        else if(SC_1_enable)
                SC_2_enable <= 1;
        else
                SC_2_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_3 <= 0;
        else if(SC_4_enable)
                SC_3 <= SC_3 + 1;
        else
                SC_3 <= SC_3;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_read[3] <= 0;
        else if(SC_read[2])
```

```
                    SC_read[3] <= 1;
          else
                    SC_read[3] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
          if(rst_SC_1_counter)
                    SC_3_enable <= 0;
          else if(SC_2_enable)
                    SC_3_enable <= 1;
          else
                    SC_3_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
          if(rst_SC_1_counter)
                    SC_4 <= 0;
          else if(SC_5_enable)
                    SC_4 <= SC_4 + 1;
          else
                    SC_4 <= SC_4;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
          if(rst_SC_1_counter)
                    SC_read[4] <= 0;
          else if(SC_read[3])
                    SC_read[4] <= 1;
          else
                    SC_read[4] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
          if(rst_SC_1_counter)
                    SC_4_enable <= 0;
          else if(SC_3_enable)
                    SC_4_enable <= 1;
          else
                    SC_4_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
          if(rst_SC_1_counter)
                    SC_5 <= 0;
          else if(SC_6_enable)
                    SC_5 <= SC_5 + 1;
          else
                    SC_5 <= SC_5;
end

always@(posedge INT or posedge rst_SC_1_counter)
```

```
begin
        if(rst_SC_1_counter)
                SC_read[5] <= 0;
        else if(SC_read[4])
                SC_read[5] <= 1;
        else
                SC_read[5] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_5_enable <= 0;
        else if(SC_4_enable)
                SC_5_enable <= 1;
        else
                SC_5_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_6 <= 0;
        else if(SC_7_enable)
                SC_6 <= SC_6 + 1;
        else
                SC_6 <= SC_6;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_read[6] <= 0;
        else if(SC_read[5])
                SC_read[6] <= 1;
        else
                SC_read[6] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_6_enable <= 0;
        else if(SC_5_enable)
                SC_6_enable <= 1;
        else
                SC_6_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_7 <= 0;
        else if(SC_8_enable)
                SC_7 <= SC_7 + 1;
        else
```

```
				SC_7 <= SC_7;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
		if(rst_SC_1_counter)
				SC_read[7] <= 0;
		else if(SC_read[6])
				SC_read[7] <= 1;
		else
				SC_read[7] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
		if(rst_SC_1_counter)
				SC_7_enable <= 0;
		else if(SC_6_enable)
				SC_7_enable <= 1;
		else
				SC_7_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
		if(rst_SC_1_counter)
				SC_8 <= 0;
		else if(SC_9_enable)
				SC_8 <= SC_8 + 1;
		else
				SC_8 <= SC_8;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
		if(rst_SC_1_counter)
				SC_read[8] <= 0;
		else if(SC_read[7])
				SC_read[8] <= 1;
		else
				SC_read[8] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
		if(rst_SC_1_counter)
				SC_8_enable <= 0;
		else if(SC_7_enable)
				SC_8_enable <= 1;
		else
				SC_8_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
		if(rst_SC_1_counter)
```

```
                SC_9 <= 0;
        else if(SC_10_enable)
                SC_9 <= SC_9 + 1;
        else
                SC_9 <= SC_9;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_read[9] <= 0;
        else if(SC_read[8])
                SC_read[9] <= 1;
        else
                SC_read[9] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_9_enable <= 0;
        else if(SC_8_enable)
                SC_9_enable <= 1;
        else
                SC_9_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_10 <= 0;
        else if(SC_11_enable)
                SC_10 <= SC_10 + 1;
        else
                SC_10 <= SC_10;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_read[10] <= 0;
        else if(SC_read[9])
                SC_read[10] <= 1;
        else
                SC_read[10] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_10_enable <= 0;
        else if(SC_9_enable)
                SC_10_enable <= 1;
        else
                SC_10_enable <= 0;
end
```

```
always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_11 <= 0;
        else if(SC_12_enable)
                SC_11 <= SC_11 + 1;
        else
                SC_11 <= SC_11;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_11_enable <= 0;
        else if(SC_10_enable)
                SC_11_enable <= 1;
        else
                SC_11_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_read[11] <= 0;
        else if(SC_read[10])
                SC_read[11] <= 1;
        else
                SC_read[11] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_12 <= 0;
        else if(SC_13_enable)
                SC_12 <= SC_12 + 1;
        else
                SC_12 <= SC_12;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_read[12] <= 0;
        else if(SC_read[11])
                SC_read[12] <= 1;
        else
                SC_read[12] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_12_enable <= 0;
        else if(SC_11_enable)
```

```
                SC_12_enable <= 1;
        else
                SC_12_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_13 <= 0;
        else if(SC_14_enable)
                SC_13 <= SC_13 + 1;
        else
                SC_13 <= SC_13;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_read[13] <= 0;
        else if(SC_read[12])
                SC_read[13] <= 1;
        else
                SC_read[13] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_13_enable <= 0;
        else if(SC_12_enable)
                SC_13_enable <= 1;
        else
                SC_13_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_14 <= 0;
        else if(SC_15_enable)
                SC_14 <= SC_14 + 1;
        else
                SC_14 <= SC_14;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_read[14] <= 0;
        else if(SC_read[13])
                SC_read[14] <= 1;
        else
                SC_read[14] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
```

```
begin
        if(rst_SC_1_counter)
                SC_14_enable <= 0;
        else if(SC_13_enable)
                SC_14_enable <= 1;
        else
                SC_14_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_15 <= 0;
        else if(SC_16_enable)
                SC_15 <= SC_15 + 1;
        else
                SC_15 <= SC_15;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_read[15] <= 0;
        else if(SC_read[14])
                SC_read[15] <= 1;
        else
                SC_read[15] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_15_enable <= 0;
        else if(SC_14_enable)
                SC_15_enable <= 1;
        else
                SC_15_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_16 <= 0;
        else if(SC_17_enable)
                SC_16 <= SC_16 + 1;
        else
                SC_16 <= SC_16;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
        if(rst_SC_1_counter)
                SC_read[16] <= 0;
        else if(SC_read[15])
                SC_read[16] <= 1;
        else
```

```
			SC_read[16] <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
		if(rst_SC_1_counter)
			SC_16_enable <= 0;
		else if(SC_15_enable)
			SC_16_enable <= 1;
		else
			SC_16_enable <= 0;
end

always@(posedge INT or posedge rst_SC_1_counter)
begin
		if(rst_SC_1_counter)
			SC_17_enable <= 0;
		else if(SC_16_enable)
			SC_17_enable <= 1;
		else
			SC_17_enable <= 0;
end

assign SC_read_final[1]=SC_read[1] | rst;
assign SC_read_final[2]=SC_read[2] | rst;
assign SC_read_final[3]=SC_read[3] | rst;
assign SC_read_final[4]=SC_read[4] | rst;
assign SC_read_final[5]=SC_read[5] | rst;
assign SC_read_final[6]=SC_read[6] | rst;
assign SC_read_final[7]=SC_read[7] | rst;
assign SC_read_final[8]=SC_read[8] | rst;
assign SC_read_final[9]=SC_read[9] | rst;
assign SC_read_final[10]=SC_read[10] | rst;
assign SC_read_final[11]=SC_read[11] | rst;
assign SC_read_final[12]=SC_read[12] | rst;
assign SC_read_final[13]=SC_read[13] | rst;
assign SC_read_final[14]=SC_read[14] | rst;
assign SC_read_final[15]=SC_read[15] | rst;
assign SC_read_final[16]=SC_read[16] | rst;


assign SC_1_RST=(SC_read[1] & RST_counter) | rst;
assign SC_2_RST=(SC_read[2] & RST_counter) | rst;
assign SC_3_RST=(SC_read[3] & RST_counter) | rst;
assign SC_4_RST=(SC_read[4] & RST_counter) | rst;
assign SC_5_RST=(SC_read[5] & RST_counter) | rst;
assign SC_6_RST=(SC_read[6] & RST_counter) | rst;
assign SC_7_RST=(SC_read[7] & RST_counter) | rst;
assign SC_8_RST=(SC_read[8] & RST_counter) | rst;
assign SC_9_RST=(SC_read[9] & RST_counter) | rst;
assign SC_10_RST=(SC_read[10] & RST_counter) | rst;
assign SC_11_RST=(SC_read[11] & RST_counter) | rst;
assign SC_12_RST=(SC_read[12] & RST_counter) | rst;
assign SC_13_RST=(SC_read[13] & RST_counter) | rst;
assign SC_14_RST=(SC_read[14] & RST_counter) | rst;
assign SC_15_RST=(SC_read[15] & RST_counter) | rst;
assign SC_16_RST=(SC_read[16] & RST_counter) | rst;

endmodule

`timescale 1ns/1ps
module serial_interface( clk , rst , enable,  serial_in , reg_out);

input enable, clk, rst, serial_in;

output [54:0] reg_out;

reg [54:0] reg_out;


always@(posedge clk or posedge rst)
begin
        if(rst)
                reg_out<=0;
        else
                if(enable)
                begin
                reg_out[0]<=serial_in;
                reg_out[54:1]<=reg_out[53:0];
                end
                else
                reg_out[54:0]<=reg_out[54:0];

end

endmodule


// control reg module//
//**********************//
//**********************//
`timescale 1ns/1ps
module control_register(clk , rst , enable, serial_transfer ,parallel_data, control_reg, shift_reg);

input clk,rst,enable,serial_transfer;
input[54:0] shift_reg;
input[9:0] parallel_data;
output[54:0] control_reg;
reg [54:0] control_reg;

always@(posedge clk or posedge rst)
begin
        if(rst)
                control_reg<=0;
        else
                if(serial_transfer && !enable)//serial mode
                begin
                        if(shift_reg[42]==1) //PGM
                        begin

                                if(shift_reg[54]==1)//CNT
                                        control_reg[54:0]<=shift_reg[54:0];//allow transfer
                                else
                                        control_reg[54:43]<=control_reg[54:43];//keep previous

                                        control_reg[42:0]<=shift_reg[42:0];//allow transfer

                        end
                        else
                                control_reg[54:0]<=control_reg[54:0];

                end
                else//parallel mode
                begin
                        control_reg[53:44]<=parallel_data[9:0]; //parallel data
                        control_reg[43]<=1;//all pixels_selected
                        control_reg[54]<= control_reg[54]; //CNT
                        control_reg[42:0]<=control_reg[42:0]; //PGM, data, address, int time and res time
remain same
                end
end

endmodule


//control_signals module//
//**********************//
//**********************//
`timescale 1ns/1ps
module control_signals (clk, control_reg, TDI_pos_y, int_time, res_time,
                        BYPASS1, BYPASS2, BYPASS3, BYPASS4, BYPASS5, BYPASS6, BYPASS7,
BYPASS8, TDI);
input clk;
input [54:0]control_reg;
//input[6:0] cycle_count;
output TDI_pos_y;
output BYPASS1,BYPASS2, BYPASS3, BYPASS4,BYPASS5,BYPASS6,BYPASS7, BYPASS8, TDI;
//output [6:0] pixel_adress;
output[17:0] int_time;
output[8:0] res_time;

reg TDI_pos_y;
reg BYPASS1, BYPASS2, BYPASS3, BYPASS4, BYPASS5, BYPASS6, BYPASS7, BYPASS8, TDI;
reg [6:0] pixel_adress;
reg [17:0] int_time;
reg [8:0] res_time;

always@(control_reg[53:49]) //BYPASS3,BYPASS2 and BYPASS1
begin
        if(control_reg[53:49]==5'b 00000) //TDI Pos + y
        begin
                TDI_pos_y=0;
                TDI=1;
                BYPASS1=1;
                BYPASS2=1;
                BYPASS3=1;
                BYPASS4=1;
                BYPASS5=1;
```

```
		BYPASS6=1;
		BYPASS7=1;
		BYPASS8=1;
end
else if(control_reg[53:49]==5'b 00001) //BYPASS DET 1
begin
		TDI_pos_y=0;
		TDI=0;
		BYPASS1=1;
		BYPASS2=0;
		BYPASS3=0;
		BYPASS4=0;
		BYPASS5=0;
		BYPASS6=0;
		BYPASS7=0;
		BYPASS8=0;

end
else if(control_reg[53:49]==5'b 00010) //BYPASS DET 2
begin
		TDI_pos_y=0;
		TDI=0;
		BYPASS1=0;
		BYPASS2=1;
		BYPASS3=0;
		BYPASS4=0;
		BYPASS5=0;
		BYPASS6=0;
		BYPASS7=0;
		BYPASS8=0;
end
else if(control_reg[53:49]==5'b 00011) //BYPASS DET 3
begin
		TDI_pos_y=0;
		TDI=0;
		BYPASS1=0;
		BYPASS2=0;
		BYPASS3=1;
		BYPASS4=0;
		BYPASS5=0;
		BYPASS6=0;
		BYPASS7=0;
		BYPASS8=0;
end
else if(control_reg[53:49]==5'b 00100) //BYPASS DET 4
begin
		TDI_pos_y=0;
		TDI=0;
		BYPASS1=0;
		BYPASS2=0;
		BYPASS3=0;
		BYPASS4=1;
		BYPASS5=0;
		BYPASS6=0;
		BYPASS7=0;
		BYPASS8=0;
```

```
end


else if(control_reg[53:49]==5'b 10000) //TDI neg - y
begin
        TDI_pos_y=1;
        TDI=1;
        BYPASS1=0;
        BYPASS2=0;
        BYPASS3=0;
        BYPASS4=0;
        BYPASS5=0;
        BYPASS6=0;
        BYPASS7=0;
        BYPASS8=0;
end

else if(control_reg[53:49]==5'b 10001) //BYPASS DET 5
begin
        TDI_pos_y=0;
        TDI=0;
        BYPASS1=0;
        BYPASS2=0;
        BYPASS3=0;
        BYPASS4=0;
        BYPASS5=1;
        BYPASS6=0;
        BYPASS7=0;
        BYPASS8=0;
end

else if(control_reg[53:49]==5'b 10010) //BYPASS DET 6
begin
        TDI_pos_y=0;
        TDI=0;
        BYPASS1=0;
        BYPASS2=0;
        BYPASS3=0;
        BYPASS4=0;
        BYPASS5=0;
        BYPASS6=1;
        BYPASS7=0;
        BYPASS8=0;
end

else if(control_reg[53:49]==5'b 10011) //BYPASS DET 7
begin
        TDI_pos_y=0;
        TDI=0;
        BYPASS1=0;
        BYPASS2=0;
        BYPASS3=0;
        BYPASS4=0;
        BYPASS5=0;
        BYPASS6=0;
        BYPASS7=1;
```

```
				BYPASS8=0;
			end

			else if(control_reg[53:49]==5'b 10100) //BYPASS DET 8
			begin
				TDI_pos_y=0;
				TDI=0;
				BYPASS1=0;
				BYPASS2=0;
				BYPASS3=0;
				BYPASS4=0;
				BYPASS5=0;
				BYPASS6=0;
				BYPASS7=0;
				BYPASS8=1;
			end

			else //not allowed
			begin
				TDI_pos_y=0;
				TDI=1;
				BYPASS1=1;
				BYPASS2=1;
				BYPASS3=1;
				BYPASS4=1;
				BYPASS5=1;
				BYPASS6=1;
				BYPASS7=1;
				BYPASS8=1;
			end
end


always@(control_reg[54:42])
begin
			if (control_reg[48:46] == 3'b000)
				int_time <= control_reg[26:9];
			else if (control_reg[48:46] == 3'b001)
				int_time <= 18'b 00_0001_0000_0000_0000;
			else if (control_reg[48:46] == 3'b010)
				int_time <= 18'b 00_0000_1100_0000_0000;
			else if (control_reg[48:46] == 3'b011)
				int_time <= 18'b 00_0000_1000_0000_0000;
			else if (control_reg[48:46] == 3'b100)
				int_time <= 18'b 00_0000_0110_0000_0000;
			else if (control_reg[48:46] == 3'b101)
				int_time <= 18'b 00_0000_0100_0000_0000;
			else if (control_reg[48:46] == 3'b110)
				int_time <= 18'b 00_0000_0010_0000_0000;
			else
				int_time <= 18'b 00_0000_0001_0000_0000;
end

always@(control_reg[54:42])
begin
			if (control_reg[45:44] == 2'b00)
```

```
                res_time <= control_reg[8:0];
        else if (control_reg[45:44] == 2'b01)
                res_time <= 9'b 1_0000_0000;
        else if (control_reg[45:44] == 2'b10)
                res_time <= 9'b 0_1000_0000;
        else
                res_time <= 9'b 0_0100_0000;


end

endmodule


`timescale 1ns/1ps
module INT (clk, rst, enable, int_time, res_time, INT, INT_counter, read_counter, RST_CAP, RST_RAMP,
RST_counter, read, load, residue);

input clk, rst, enable;
input [17:0] int_time;
input [8:0] res_time;
output INT;
output [18:0] INT_counter;
output [12:0] read_counter;
output RST_CAP, RST_RAMP, RST_counter, residue;
output read, load;

reg [18:0] INT_counter;
reg INT;
reg RST_CAP, RST_RAMP, RST_counter, residue;
reg read, load;
reg [12:0] read_counter;

wire rst_INT;

assign rst_INT = rst | enable;

always@(posedge clk or posedge rst_INT)
begin
        if(rst_INT)
                begin
                RST_CAP <= 1;
                end

        else if(INT_counter > (int_time + res_time + 22) && INT_counter < (int_time + res_time + 36))
                begin
                RST_CAP <= 1;
                end
        else if(INT_counter >= (int_time + res_time + 36) && INT_counter < 2879)
                begin
                RST_CAP <= 1;
                end
        else
                begin
                RST_CAP <= 0;
                end
```

```
end

always@(posedge clk or posedge rst)
begin
        if(rst)
                residue <= 1;
        else if(INT_counter > (int_time + 22) && INT_counter < (int_time + res_time + 23))
                residue <= 1;
        else
                residue <= 0;
end

always@(posedge clk or posedge rst_INT)
begin
        if(rst_INT)
                begin
                RST_RAMP <= 0;
                end
        else if(INT_counter < int_time + 23 && INT_counter >= int_time + 1)
                begin
                RST_RAMP <= 1;
                end

        else
                begin
                RST_RAMP <= 0;
                end
end

always@(posedge clk or posedge rst_INT)
begin
        if(rst_INT)
                begin
                RST_counter <= 1;
                end
        else if(INT_counter > (int_time + res_time + 22) && INT_counter < (int_time + res_time + 36) &&
INT_counter >= 2879)
                begin
                RST_counter <= 1;
                end
        else if(INT_counter >= 2879 && INT_counter > (int_time + res_time + 36) && INT_counter <=
2883)
                begin
                RST_counter <= 1;
                end
        else
                begin
                RST_counter <= 0;
                end
end

always@(posedge clk or posedge rst_INT)
begin
        if(rst_INT)
                begin
                INT_counter <= 0;
                INT <= 0;
                end
        else if(INT_counter == (res_time + int_time + 36) && INT_counter >= 2883)
                begin
                INT_counter <= 0;
                INT <= 1;
                end
        else if(INT_counter >= int_time && INT_counter <= (res_time + int_time + 36))
                begin
                INT_counter <= INT_counter+1;
                INT <= 0;
                end
        else if(INT_counter >= int_time && INT_counter < 2884)
                begin
                INT_counter <= INT_counter+1;
                INT <= 0;
                end
        else if(INT_counter < int_time)
                begin
                INT_counter <= INT_counter+1;
                INT <= 1;
                end
        else
                begin
                INT_counter <= 0;
                INT <= 0;
                end
end

always@(posedge clk or posedge rst_INT)
begin
        if(rst_INT)
                begin
                read_counter <= 0;
                read <= 0;
                end
        else if (INT_counter >= 2879)
                begin
                read_counter <= 0;
                read <= 0;
                end
        else
                begin
                read_counter <= read_counter + 1;
                read <= 1;
                end
end

always@(posedge clk or posedge rst_INT)
begin
        if(rst_INT)
                begin
                load <= 0;
                end
        else if (read_counter[4:0] == 5'b 00010 || read_counter[4:0] == 5'b 00001)
                begin
```

```
                        load <= 1;
                        end
            else
                        begin
                        load <= 0;
                        end
end


endmodule

`timescale 1ns/1ps
module toplevel (clk, rst, enable, serial_in, serial_transfer, parallel_data,
            INT, BYPASS1, BYPASS2, BYPASS3, BYPASS4, BYPASS5, BYPASS6, BYPASS7, BYPASS8,
TDI_pos_y,
            SC_1_RST_MSB_N, SC_1_RST_LSB_N, SC_1_MSB_enable, SC_1_LSB_enable, SC_1,
SC_1_LSB,
            SC_2_RST_MSB_N, SC_2_RST_LSB_N, SC_2_MSB_enable, SC_2_LSB_enable, SC_2,
SC_2_LSB,
            SC_3_RST_MSB_N, SC_3_RST_LSB_N, SC_3_MSB_enable, SC_3_LSB_enable, SC_3,
SC_3_LSB,
            SC_4_RST_MSB_N, SC_4_RST_LSB_N, SC_4_MSB_enable, SC_4_LSB_enable, SC_4,
SC_4_LSB,
            SC_5_RST_MSB_N, SC_5_RST_LSB_N, SC_5_MSB_enable, SC_5_LSB_enable, SC_5,
SC_5_LSB,
            SC_6_RST_MSB_N, SC_6_RST_LSB_N, SC_6_MSB_enable, SC_6_LSB_enable, SC_6,
SC_6_LSB,
            SC_7_RST_MSB_N, SC_7_RST_LSB_N, SC_7_MSB_enable, SC_7_LSB_enable, SC_7,
SC_7_LSB,
            SC_8_RST_MSB_N, SC_8_RST_LSB_N, SC_8_MSB_enable, SC_8_LSB_enable, SC_8,
SC_8_LSB,
            SC_9_RST_MSB_N, SC_9_RST_LSB_N, SC_9_MSB_enable, SC_9_LSB_enable, SC_9,
SC_9_LSB,
            SC_10_RST_MSB_N, SC_10_RST_LSB_N, SC_10_MSB_enable, SC_10_LSB_enable, SC_10,
SC_10_LSB,
            SC_11_RST_MSB_N, SC_11_RST_LSB_N, SC_11_MSB_enable, SC_11_LSB_enable, SC_11,
SC_11_LSB,
            SC_12_RST_MSB_N, SC_12_RST_LSB_N, SC_12_MSB_enable, SC_12_LSB_enable, SC_12,
SC_12_LSB,
            SC_13_RST_MSB_N, SC_13_RST_LSB_N, SC_13_MSB_enable, SC_13_LSB_enable, SC_13,
SC_13_LSB,
            SC_14_RST_MSB_N, SC_14_RST_LSB_N, SC_14_MSB_enable, SC_14_LSB_enable, SC_14,
SC_14_LSB,
            SC_15_RST_MSB_N, SC_15_RST_LSB_N, SC_15_MSB_enable, SC_15_LSB_enable, SC_15,
SC_15_LSB,
            SC_16_RST_MSB_N, SC_16_RST_LSB_N, SC_16_MSB_enable, SC_16_LSB_enable, SC_16,
SC_16_LSB,
            RST_CAP, RST_RAMP, load, residue, SC_read_final, SC_read_LSB, row_0_31, row_32_63,
row_64_95);

input clk, rst, enable, serial_in, serial_transfer;
input [9:0] parallel_data;

output BYPASS1, BYPASS2, BYPASS3, BYPASS4, BYPASS5, BYPASS6, BYPASS7, BYPASS8,
TDI_pos_y;
output INT;
output load;
output RST_CAP, RST_RAMP, residue;

output SC_1_MSB_enable, SC_1_LSB_enable;
output SC_2_MSB_enable, SC_2_LSB_enable;
output SC_3_MSB_enable, SC_3_LSB_enable;
output SC_4_MSB_enable, SC_4_LSB_enable;
output SC_5_MSB_enable, SC_5_LSB_enable;
output SC_6_MSB_enable, SC_6_LSB_enable;
output SC_7_MSB_enable, SC_7_LSB_enable;
output SC_8_MSB_enable, SC_8_LSB_enable;
output SC_9_MSB_enable, SC_9_LSB_enable;
output SC_10_MSB_enable, SC_10_LSB_enable;
output SC_11_MSB_enable, SC_11_LSB_enable;
output SC_12_MSB_enable, SC_12_LSB_enable;
output SC_13_MSB_enable, SC_13_LSB_enable;
output SC_14_MSB_enable, SC_14_LSB_enable;
output SC_15_MSB_enable, SC_15_LSB_enable;
output SC_16_MSB_enable, SC_16_LSB_enable;

output SC_1_RST_MSB_N, SC_1_RST_LSB_N;
output SC_2_RST_MSB_N, SC_2_RST_LSB_N;
output SC_3_RST_MSB_N, SC_3_RST_LSB_N;
output SC_4_RST_MSB_N, SC_4_RST_LSB_N;
output SC_5_RST_MSB_N, SC_5_RST_LSB_N;
output SC_6_RST_MSB_N, SC_6_RST_LSB_N;
output SC_7_RST_MSB_N, SC_7_RST_LSB_N;
output SC_8_RST_MSB_N, SC_8_RST_LSB_N;
output SC_9_RST_MSB_N, SC_9_RST_LSB_N;
output SC_10_RST_MSB_N, SC_10_RST_LSB_N;
output SC_11_RST_MSB_N, SC_11_RST_LSB_N;
output SC_12_RST_MSB_N, SC_12_RST_LSB_N;
output SC_13_RST_MSB_N, SC_13_RST_LSB_N;
output SC_14_RST_MSB_N, SC_14_RST_LSB_N;
output SC_15_RST_MSB_N, SC_15_RST_LSB_N;
output SC_16_RST_MSB_N, SC_16_RST_LSB_N;

output [16:1] SC_read_final, SC_read_LSB;
output [2:0] SC_1, SC_1_LSB;
output [2:0] SC_2, SC_2_LSB;
output [2:0] SC_3, SC_3_LSB;
output [2:0] SC_4, SC_4_LSB;
output [2:0] SC_5, SC_5_LSB;
output [2:0] SC_6, SC_6_LSB;
output [2:0] SC_7, SC_7_LSB;
output [2:0] SC_8, SC_8_LSB;
output [2:0] SC_9, SC_9_LSB;
output [2:0] SC_10, SC_10_LSB;
output [2:0] SC_11, SC_11_LSB;
output [2:0] SC_12, SC_12_LSB;
output [2:0] SC_13, SC_13_LSB;
output [2:0] SC_14, SC_14_LSB;
output [2:0] SC_15, SC_15_LSB;
output [2:0] SC_16, SC_16_LSB;

output [31:0] row_0_31;
output [31:0] row_32_63;
output [31:0] row_64_95;

wire BYPASS1, BYPASS2, BYPASS3, BYPASS4, BYPASS5, BYPASS6, BYPASS7, BYPASS8, TDI,
TDI_pos_y;
wire MSB, LSB;
wire read, load;
wire INT;
wire INT_BAR;
wire rst_SC, rst_SC_counter, rst_SC_1_counter;
wire RST_CAP, RST_RAMP, RST_counter, residue;
wire [18:0] INT_counter;
wire [12:0] read_counter;
wire [54:0] shift_reg_out;
wire [54:0] control_reg;
wire [17:0] int_time;
wire [8:0] res_time;

wire [1:0] SC_1_counter;

wire SC_1_enable;
wire SC_2_enable;
wire SC_3_enable;
wire SC_4_enable;
wire SC_5_enable;
wire SC_6_enable;
wire SC_7_enable;
wire SC_8_enable;
wire SC_9_enable;
wire SC_10_enable;
wire SC_11_enable;
wire SC_12_enable;
wire SC_13_enable;
wire SC_14_enable;
wire SC_15_enable;
wire SC_16_enable;

wire SC_1_MSB_enable, SC_1_LSB_enable;
wire SC_2_MSB_enable, SC_2_LSB_enable;
wire SC_3_MSB_enable, SC_3_LSB_enable;
wire SC_4_MSB_enable, SC_4_LSB_enable;
wire SC_5_MSB_enable, SC_5_LSB_enable;
wire SC_6_MSB_enable, SC_6_LSB_enable;
wire SC_7_MSB_enable, SC_7_LSB_enable;
wire SC_8_MSB_enable, SC_8_LSB_enable;
wire SC_9_MSB_enable, SC_9_LSB_enable;
wire SC_10_MSB_enable, SC_10_LSB_enable;
wire SC_11_MSB_enable, SC_11_LSB_enable;
wire SC_12_MSB_enable, SC_12_LSB_enable;
wire SC_13_MSB_enable, SC_13_LSB_enable;
wire SC_14_MSB_enable, SC_14_LSB_enable;
wire SC_15_MSB_enable, SC_15_LSB_enable;
wire SC_16_MSB_enable, SC_16_LSB_enable;

wire SC_1_RST;
wire SC_2_RST;
wire SC_3_RST;
wire SC_4_RST;
wire SC_5_RST;
wire SC_6_RST;
wire SC_7_RST;
wire SC_8_RST;
wire SC_9_RST;
wire SC_10_RST;
wire SC_11_RST;
wire SC_12_RST;
wire SC_13_RST;
wire SC_14_RST;
wire SC_15_RST;
wire SC_16_RST;

wire SC_1_RST_MSB_N, SC_1_RST_LSB_N;
wire SC_2_RST_MSB_N, SC_2_RST_LSB_N;
wire SC_3_RST_MSB_N, SC_3_RST_LSB_N;
wire SC_4_RST_MSB_N, SC_4_RST_LSB_N;
wire SC_5_RST_MSB_N, SC_5_RST_LSB_N;
wire SC_6_RST_MSB_N, SC_6_RST_LSB_N;
wire SC_7_RST_MSB_N, SC_7_RST_LSB_N;
wire SC_8_RST_MSB_N, SC_8_RST_LSB_N;
wire SC_9_RST_MSB_N, SC_9_RST_LSB_N;
wire SC_10_RST_MSB_N, SC_10_RST_LSB_N;
wire SC_11_RST_MSB_N, SC_11_RST_LSB_N;
wire SC_12_RST_MSB_N, SC_12_RST_LSB_N;
wire SC_13_RST_MSB_N, SC_13_RST_LSB_N;
wire SC_14_RST_MSB_N, SC_14_RST_LSB_N;
wire SC_15_RST_MSB_N, SC_15_RST_LSB_N;
wire SC_16_RST_MSB_N, SC_16_RST_LSB_N;

wire [16:1] SC_read, SC_read_final, SC_read_LSB;

wire [2:0] SC_1, SC_1_LSB;
wire [2:0] SC_2, SC_2_LSB;
wire [2:0] SC_3, SC_3_LSB;
wire [2:0] SC_4, SC_4_LSB;
wire [2:0] SC_5, SC_5_LSB;
wire [2:0] SC_6, SC_6_LSB;
wire [2:0] SC_7, SC_7_LSB;
wire [2:0] SC_8, SC_8_LSB;
wire [2:0] SC_9, SC_9_LSB;
wire [2:0] SC_10, SC_10_LSB;
wire [2:0] SC_11, SC_11_LSB;
wire [2:0] SC_12, SC_12_LSB;
wire [2:0] SC_13, SC_13_LSB;
wire [2:0] SC_14, SC_14_LSB;
wire [2:0] SC_15, SC_15_LSB;
wire [2:0] SC_16, SC_16_LSB;

wire [3:0] dec_out;
wire [31:0] row_0_31;
wire [31:0] row_32_63;
wire [31:0] row_64_95;
```

```
INT INT_1(clk, rst, enable, int_time, res_time, INT, INT_counter, read_counter, RST_CAP, RST_RAMP,
RST_counter, read, load, residue);
serial_interface interface( clk , rst , enable,  serial_in , shift_reg_out);
control_register ctrl_reg(clk , rst , enable, serial_transfer ,parallel_data, control_reg, shift_reg_out);
control_signals ctrl_signals(clk, control_reg, TDI_pos_y, int_time, res_time, BYPASS1, BYPASS2,
BYPASS3, BYPASS4, BYPASS5, BYPASS6, BYPASS7, BYPASS8, TDI);

dec2to4 row_dec_main(read, read_counter[11:10], dec_out);
dec5to32 row_dec_1(dec_out[0], read_counter[9:5], row_0_31);
dec5to32 row_dec_2(dec_out[1], read_counter[9:5], row_32_63);
dec5to32 row_dec_3(dec_out[2], read_counter[9:5], row_64_95);


SC SC_mux (clk, INT, rst, RST_counter, rst_SC, rst_SC_counter, rst_SC_1_counter, SC_1_RST,
SC_1_enable, SC_1, SC_1_counter,
            SC_2_RST, SC_2_enable, SC_2, SC_3_RST, SC_3_enable, SC_3, SC_4_RST, SC_4_enable,
SC_4,
            SC_5_RST, SC_5_enable, SC_5, SC_6_RST, SC_6_enable, SC_6, SC_7_RST, SC_7_enable,
SC_7,
            SC_8_RST, SC_8_enable, SC_8, SC_9_RST, SC_9_enable, SC_9, SC_10_RST, SC_10_enable,
SC_10,
            SC_11_RST, SC_11_enable, SC_11, SC_12_RST, SC_12_enable, SC_12, SC_13_RST,
SC_13_enable, SC_13,
            SC_14_RST, SC_14_enable, SC_14, SC_15_RST, SC_15_enable, SC_15, SC_16_RST,
SC_16_enable, SC_16, SC_read, SC_read_final);

assign INT_BAR=~INT;
assign MSB=INT;
assign LSB=INT_BAR;

assign SC_1_MSB_enable = (SC_1_enable & MSB) | rst;
assign SC_1_LSB_enable = (SC_1_enable & LSB) | rst;
assign SC_2_MSB_enable = (SC_2_enable & MSB) | rst;
assign SC_2_LSB_enable = (SC_2_enable & LSB) | rst;
assign SC_3_MSB_enable = (SC_3_enable & MSB) | rst;
assign SC_3_LSB_enable = (SC_3_enable & LSB) | rst;
assign SC_4_MSB_enable = (SC_4_enable & MSB) | rst;
assign SC_4_LSB_enable = (SC_4_enable & LSB) | rst;
assign SC_5_MSB_enable = (SC_5_enable & MSB) | rst;
assign SC_5_LSB_enable = (SC_5_enable & LSB) | rst;
assign SC_6_MSB_enable = (SC_6_enable & MSB) | rst;
assign SC_6_LSB_enable = (SC_6_enable & LSB) | rst;
assign SC_7_MSB_enable = (SC_7_enable & MSB) | rst;
assign SC_7_LSB_enable = (SC_7_enable & LSB) | rst;
assign SC_8_MSB_enable = (SC_8_enable & MSB) | rst;
assign SC_8_LSB_enable = (SC_8_enable & LSB) | rst;
assign SC_9_MSB_enable = (SC_9_enable & MSB) | rst;
assign SC_9_LSB_enable = (SC_9_enable & LSB) | rst;
assign SC_10_MSB_enable = (SC_10_enable & MSB) | rst;
assign SC_10_LSB_enable = (SC_10_enable & LSB) | rst;
assign SC_11_MSB_enable = (SC_11_enable & MSB) | rst;
assign SC_11_LSB_enable = (SC_11_enable & LSB) | rst;
assign SC_12_MSB_enable = (SC_12_enable & MSB) | rst;
assign SC_12_LSB_enable = (SC_12_enable & LSB) | rst;
assign SC_13_MSB_enable = (SC_13_enable & MSB) | rst;
assign SC_13_LSB_enable = (SC_13_enable & LSB) | rst;
assign SC_14_MSB_enable = (SC_14_enable & MSB) | rst;
assign SC_14_LSB_enable = (SC_14_enable & LSB) | rst;
assign SC_15_MSB_enable = (SC_15_enable & MSB) | rst;
assign SC_15_LSB_enable = (SC_15_enable & LSB) | rst;
assign SC_16_MSB_enable = (SC_16_enable & MSB) | rst;
assign SC_16_LSB_enable = (SC_16_enable & LSB) | rst;

assign SC_1_RST_MSB_N=~SC_1_RST;
assign SC_2_RST_MSB_N=~SC_2_RST;
assign SC_3_RST_MSB_N=~SC_3_RST;
assign SC_4_RST_MSB_N=~SC_4_RST;
assign SC_5_RST_MSB_N=~SC_5_RST;
assign SC_6_RST_MSB_N=~SC_6_RST;
assign SC_7_RST_MSB_N=~SC_7_RST;
assign SC_8_RST_MSB_N=~SC_8_RST;
assign SC_9_RST_MSB_N=~SC_9_RST;
assign SC_10_RST_MSB_N=~SC_10_RST;
assign SC_11_RST_MSB_N=~SC_11_RST;
assign SC_12_RST_MSB_N=~SC_12_RST;
assign SC_13_RST_MSB_N=~SC_13_RST;
assign SC_14_RST_MSB_N=~SC_14_RST;
assign SC_15_RST_MSB_N=~SC_15_RST;
assign SC_16_RST_MSB_N=~SC_16_RST;

assign SC_1_RST_LSB_N=SC_1_RST_MSB_N;
assign SC_2_RST_LSB_N=SC_2_RST_MSB_N;
assign SC_3_RST_LSB_N=SC_3_RST_MSB_N;
assign SC_4_RST_LSB_N=SC_4_RST_MSB_N;
assign SC_5_RST_LSB_N=SC_5_RST_MSB_N;
assign SC_6_RST_LSB_N=SC_6_RST_MSB_N;
assign SC_7_RST_LSB_N=SC_7_RST_MSB_N;
assign SC_8_RST_LSB_N=SC_8_RST_MSB_N;
assign SC_9_RST_LSB_N=SC_9_RST_MSB_N;
assign SC_10_RST_LSB_N=SC_10_RST_MSB_N;
assign SC_11_RST_LSB_N=SC_11_RST_MSB_N;
assign SC_12_RST_LSB_N=SC_12_RST_MSB_N;
assign SC_13_RST_LSB_N=SC_13_RST_MSB_N;
assign SC_14_RST_LSB_N=SC_14_RST_MSB_N;
assign SC_15_RST_LSB_N=SC_15_RST_MSB_N;
assign SC_16_RST_LSB_N=SC_16_RST_MSB_N;

assign SC_1_LSB=SC_1;
assign SC_2_LSB=SC_2;
assign SC_3_LSB=SC_3;
assign SC_4_LSB=SC_4;
assign SC_5_LSB=SC_5;
assign SC_6_LSB=SC_6;
assign SC_7_LSB=SC_7;
assign SC_8_LSB=SC_8;
assign SC_9_LSB=SC_9;
assign SC_10_LSB=SC_10;
assign SC_11_LSB=SC_11;
assign SC_12_LSB=SC_12;
assign SC_13_LSB=SC_13;
assign SC_14_LSB=SC_14;
assign SC_15_LSB=SC_15;
assign SC_16_LSB=SC_16;

assign SC_read_LSB=SC_read_final;

endmodule
```

7.1.2 Serializer

```
`timescale 1ns/1ps
module prl_to_srl_convrtr( clk , reset , enable, parallel_in , serial_out);

input [0:21] parallel_in;
input clk,reset,enable;
output serial_out;

reg [0:21] parallel_load;
reg serial_out;
reg [4:0] counter;

always@(posedge clk)
begin
        if(enable)
                counter <= 0;
        else if (counter == 21)
                counter <= counter;
        else
                counter <= counter +1;
end

always@(posedge clk or posedge reset)
begin
        if(reset)
                begin
                serial_out<=0;
                parallel_load<=0;
                end
        else if (counter == 21)
                begin
                serial_out<=0;
                parallel_load[0:21]<=parallel_in[0:21];
                end

        else

                begin
                serial_out<=parallel_load[0];
                parallel_load[0:20]<=parallel_load[1:21];
                end

end

endmodule
```