

Wire Scanner Motion Control Card

S.E. Forde

**Bergen University College - Norway
CERN – Geneva - Switzerland**

CERN Supervisor – B. Dehning AB-BI

Abstract

Scientists require a certain beam quality produced by the accelerator rings at CERN. The discovery potential of LHC is given by the reachable luminosity at its interaction points. The luminosity is maximized by minimizing the beam size. Therefore an accurate beam size measurement is required for optimizing the luminosity.

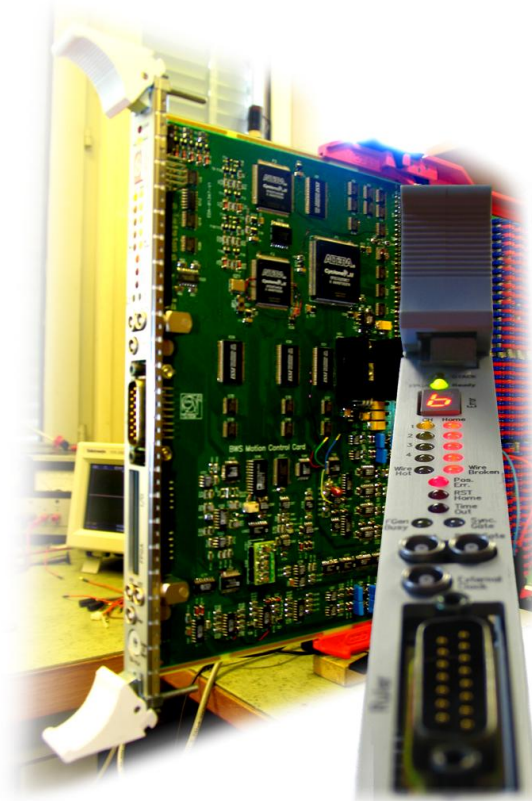
The wire scanner performs very accurate profile measurements, but as it can not be used at full intensity in the LHC ring, it is used for calibrating other profile monitors. As the current wire scanner system, which is used in the present CERN accelerators, has not been made for the required specification of the LHC, a new design of a wire scanner motion control card is part of the LHC wire scanner project. The main functions of this card are to control the wire scanner motion and to acquire the position of the wire. In case of further upgrades at a later stage, it is required to allow an easy update of the firmware, hence the programmable features of FPGAs will be used for this purpose. The FPGAs will act as the control unit of the system.

As the LHC has two separate vacuum chambers for the two counter rotating proton-beams, a wire scanner is needed for both the horizontal and vertical beam profile measurement. One motion control card is expected to control two wire scanners. The position of the wires must be acquired within a certain accuracy to meet the specification set for the LHC. In order to obtain the correct beam profile, the position acquisition must be well synchronized with the acquisition of the beam density. The values have to be stored in a memory, which is readable through the VME64x-bus.

Presented at Bergen Univ. College on 20th December 2006 – Bergen/NO

Bachelor of Science thesis

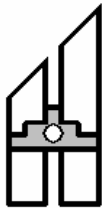
WIRE SCANNER MOTION CONTROL CARD



Stian E. S. Førde
Electronics engineering
Bergen University College
stiford@gmail.com

December 13, 2006





HØGSKOLEN I BERGEN

Avdeling for Ingeniørutdanning

TITTELBLAD FOR HOVEDPROSJEKT

<i>Rapportens tittel:</i> Wire scanner motion control card	<i>Dato:</i> 2006-12-12
<i>Forfattere:</i> Stian E. S. Førde	<i>Antall sider u/vedlegg:</i> 100
<i>Studieretning:</i> Elektronikk	<i>Antall sider vedlegg:</i> 61
<i>Vegleder ved studieretning:</i> Ketil Røed	<i>Antall CDer:</i> 1
<i>Merknader:</i>	<i>Gradering:</i> Bestått/ikkje bestått

<i>Oppdragsgiver:</i> CERN	<i>Oppdragsgivers referanse:</i>
<i>Oppdragsgivers kontaktperson:</i> Bernd Dehning	<i>Telefon:</i> +41 22 76 75 541

<i>Sammendrag:</i> LHC, den nye ring akseleratoren ved CERN, krev oppgradering av mykje utstyr for å oppnå optimale målinger. Som en del av dette må òg kontrollkortet for wire scanners oppgraderast for å forbedre nøyaktigheten på målinger av strålens kryss-størrelse. Dette kan bli oppnådd ved å introdusere større oppløysing på motor kontroll- og posisjons-data, samt utlesing av strålens tetthets-data. Det nye wire scanner kontrollkortet bruker FPGAer som kontrollenhet og fjernstyring via en VME-bus. Dette prosjektet legg vekt på det digitale designet av kontrollkortet, men berører òg dei analoge kretsane.

Stikkord:

Wire scanner	Transverse beam size	Luminosity
FPGA	VHDL	VME-bus

Høgskolen i Bergen, Avdeling for ingeniørutdanning

Postadresse: Postboks 7030, 5020 BERGEN

Tlf. 55 58 75 00

Fax 55 58 77 90

Besøksadresse: Nygårdsgaten 112, Bergen

E-post: post@hib.no Hjemmeside: <http://www.hib.no>

Acknowledgements

I have had a very rewarding time during my studies in Bergen and my time spent at CERN, and for that I would like to thank a few key people who have played a part over the past few years.

First I would like to thank my supervisor at CERN, Bernd Dehning, for providing me with the opportunity to write my thesis here by making the necessary arrangements for my Technical Studentship. Despite having a very busy schedule, he always gives priority to his staff and I am grateful for that. Also I thank my supervisor at HiB, Ketil Røed, and Solfrid Sjøstad Hasund for their cooperation and understanding during my prolonged thesis work at CERN.

During my time at CERN I have mainly worked with Jan Koopman, and I have learned a great deal from his broad experience. I especially appreciate the practical demonstrations he provided when I was in doubt and asking challenging questions to keep me on my toes.

A big thank you to the BLM-section, it has been a pleasure to spend my time with you all. Especially thanks to: Jonathan Emery for useful VHDL hints and many Labview-explanations; Ewald Effinger and Christos Zamantzas for helping me with design issues; Daniel Kramer for arranging several rendez-vous in French details; Claudine Chery for putting up with all the modifications I have asked her to make on the prototypes; and also many thanks to Markus for all the moral support.

A special thanks to all the footballers at CERN! It has been great to play with you all. Many thanks to Pit8 for accepting me on first sight, and for the after match barbeques and beers. Special thanks to Claudio Carneiro and Guy Edwards, who let me participate with the CERN team in the Association Genevoise de Football Corporatif Division 'A' Championship. It has been a pleasure to play in such a multinational team and I will never forget our victorious

moment in the cup! Thanks to the guys who joined me in the futsal team, The Mongrels, it has been good fun!

Many thanks to Theo, I appreciate the many useful discussions and traveling company. Rache and Lis, thanks for many good laughs, great food and teaching me some British slang. A big thank you to Rasmus for good advice and helping me out in many situations. Thank you Tülây, for your patient introduction in french. . . même si ma tête était dans les étoiles! Huge thank you to Håvard Hoff Langlie, Knut Ove Nygård and Bjørnar Fuglseth for the collaboration during my studies at HiB.

At last but not least, many thanks to my two wonderful sisters and my parents! I would never have taken this path without your guidance and support. Thank you for your patience and for coming by to visit when I failed to go north often enough!

Abstract

Scientists require a certain beam quality produced by the accelerator rings at CERN¹. The discovery potential of LHC² is given by the reachable luminosity at its interaction points. The luminosity is maximized by minimizing the beam size. Therefore an accurate beam size measurement is required for optimizing the luminosity.

The wire scanner performs very accurate profile measurements, but as it can not be used at full intensity in the LHC ring, it is used for calibrating other profile monitors. As the current wire scanner system, which is used in the present CERN accelerators, has not been made for the required specification of the LHC, a new design of a wire scanner motion control card is part of the LHC wire scanner project. The main functions of this card is to control the wire scanner motion and to acquire the position of the wire. In case of further upgrades at a later stage, it is required to allow an easy update of the firmware, hence the programmable features of FPGAs³ will be used for this purpose. The FPGAs will act as the control unit of the system.

As the LHC has two separate vacuum chambers for the two counter rotating proton-beams, a wire scanner is needed for both the horizontal and vertical beam profile measurement. One motion control card is expected to control two wire scanners. The position of the wires must be acquired within a certain accuracy to meet the specification set for the LHC. In order to obtain the correct beam profile, the position acquisition must be well synchronized with the acquisition of the beam density. The values have to be stored in a memory, which is readable through the VME64x-bus⁴.

Keywords: Wire scanner, luminosity, transverse beam size, FPGA, VHDL, VME

¹Conseil Européen pour la Recherche Nucléaire

²Large Hadron Collider

³Field Programmable Gate Array

⁴VERSA-Module Eurocard extended bus

Contents

1. Introduction	1
1.1. CERN	1
1.2. Particle studies at CERN	2
1.3. Why LHC	2
1.4. Luminosity	4
1.5. Wire scanners	5
1.5.1. How to control the wire scanners	8
2. Overview of the system	9
2.1. PowerPC	10
2.2. Wire scanner motion control card	10
2.3. Driver and power supply	10
2.4. High Voltage power supply and gain control	11
2.5. Digital acquisition board	11
2.6. BST - Beam Synchronization Timing	12
3. Project planning	13
3.1. Wire scanner position control	14
3.2. Acquisition	16
3.2.1. Wire scanner position acquisition	16
3.2.2. PMT and SEM acquisition	20
3.2.3. Diagnostics acquisitions	21
3.2.4. SRAM, Storing acquisition data	24
3.3. Surveillance	25
3.4. VME interface	27
4. FPGA design	29
4.1. General about FPGAs and VHDL	29
4.2. FPGA selection	30
4.3. FPGA configuration	30
4.3.1. Supported programming schemes	31
4.3.2. Which configuration scheme to use	34
4.4. FPGA master	37

4.4.1.	VME interface	38
4.4.2.	ADC and DAC signal flow control	41
4.4.3.	SRAM signal flow control	42
4.5.	FPGA slave 1	43
4.5.1.	FPGA interface control unit	44
4.5.2.	Function generator	47
4.5.3.	Function check	50
4.5.4.	Calibration	53
4.6.	FPGA slave 2	57
4.6.1.	FPGA interface control unit	57
4.6.2.	Error display	58
5.	Measurements on the prototypes	62
5.1.	FPGA prototype	62
5.1.1.	Configuration setup	63
5.1.2.	VME circuitry	64
5.1.3.	Power supplies	67
5.1.4.	Optical ruler circuitry	68
5.1.5.	SRAM circuitry	69
5.1.6.	Overall test program	71
5.2.	Prototype including analog circuits	73
5.2.1.	VMPS input (ADC0 CH0)	73
5.2.2.	Amplifier voltage check (ADC0 CH1)	73
5.2.3.	Amplifier current check (ADC0 CH2)	74
5.2.4.	Function generator loop (ADC0 CH3)	75
5.2.5.	Wire temperature circuit (ADC0 CH4 and CH5)	76
5.2.6.	Logarithmic amplifier acquisition (ADC1)	77
5.2.7.	Potentiometer input with sinusoidal signal (ADC2)	80
5.2.8.	Potentiometer accuracy (ADC2)	81
5.2.9.	Wire relay circuit	84
5.2.10.	Scope output signals	84
5.2.11.	LabVIEW testbench for WSMCC	84
5.2.12.	Controlling a wire scanner	85
6.	Conclusion	89
7.	Abbreviations, list of figures and list of tables	91
	Bibliography	99
8.	Appendix	101
A.	WSMCC functional diagram	102

B. VHDL top level block diagrams	104
B.1. Master FPGA top level	105
B.2. Slave1 FPGA top level	106
B.3. Slave2 FPGA top level	107
C. FPGA Registers	109
D. VME memory mapping	116
E. Matlab function generation script	118
F. LabVIEW VME access program	121
F.1. Front panel, user interface	121
F.2. Block diagram	122
G. LabVIEW testbench for WSMCC	123
G.1. FPGA registers check	123
G.2. ROM functions check	125
G.3. ADC0 conversion check	127
G.4. ADC1 conversion check	129
G.5. ADC2 conversion check	131
G.6. SRAM check	133
H. FPGA Prototype card	135
H.1. Schematics	135
H.2. PCB	140
H.3. FPGA pin locations	145
I. Prototype card	147
I.1. Schematics	147
I.2. PCB	159

1. Introduction

1.1. CERN

CERN is the European Organization for Nuclear Research, the world's largest particle physics centre.

CERN was founded by 12 European states in 1954. It now includes 20 member states and in total over 80 nationalities are involved. About 3000 people are employed at CERN, and another 6500 scientists do research in collaboration with CERN.

Though the main studies at CERN concern Particle Physics, it is perhaps more well known for being where the WWW¹ was invented. The WWW was born as the infrastructure of data exchange was set up in 1990. The principle of this protocol was to link related information throughout a distributed information system.

Important highlights for particle physics discoveries have also taken place at CERN. In 1957 the first accelerator was made at CERN, the Synchro-Cyclotron. Here, it was possible to observe the decay of a pion into an electron and a neutrino for the first time. One of CERN's biggest achievements was in 1973, as it was discovered, by using the PS² accelerator, that the neutrinos can interact with another particle and yet still remain as neutrinos. The SPS³ accelerator provided the first proton-antiproton collisions in 1981 and proved the existence of the W- and Z-boson in 1983. Antimatter was created at CERN in 1995, which means that a new matter is made of the antiparticles of the original matter.

¹World Wide Web

²Proton Synchrotron

³Super Proton Synchrotron

1.2. Particle studies at CERN

Ever since CERN was established, beam accelerators have been built in order to study the effect of particle collisions. For every new accelerator, the beam energy has been increased drastically. This because the higher the energy, the higher are the mass of the extracted particles and the smaller the studied distances.

Scientists have made many discoveries which have led to technological growth, yet there are many riddles left to solve. All the fundamental particles of the standard models and beyond models have still not been verified, and this will hopefully be approached by using the new accelerator, which is currently being constructed at CERN.

1.3. Why LHC

The LHC is the new cyclic accelerator currently being built at CERN, and is expected to be ready for the first injections in year 2007. It will be installed in an existing tunnel of 27 kilometers circumference, where the LEP⁴ originally was. This tunnel is located 100 meters below ground by the French-Swiss border just west of Geneva, Switzerland. Even though the tunnel is already there, all equipment must be upgraded in order to deal with the high energy stored in the LHC, there are as many as 1232 superconducting dipole magnets with their cryostats to be installed in order to bend the beam through the ring.

The LHC has two vacuum chambers where the proton- or ion-beams will be injected, one in each direction. Proton beams will be injected at 450 GeV⁵ and will be accelerated to 7 TeV⁶. This results in a total collision energy of 14 TeV, as each of the counter-rotating beams has an energy of 7 TeV.

Collisions will take place in the 4 experiment locations ALICE⁷, ATLAS⁸, CMS⁹ and LHC-B in the LHC, each experiment having its own function.

The Higgs boson is the only particle of the Standard Model which is left to be verified. It was introduced in order to be able to define the mass of other

⁴Large Electron-Positron collider

⁵Giga electron Volt

⁶Tera electron Volt

⁷A Large Ion Collider Experiment

⁸A Torodial LHC Apparatus

⁹Compact Muon Solenoid

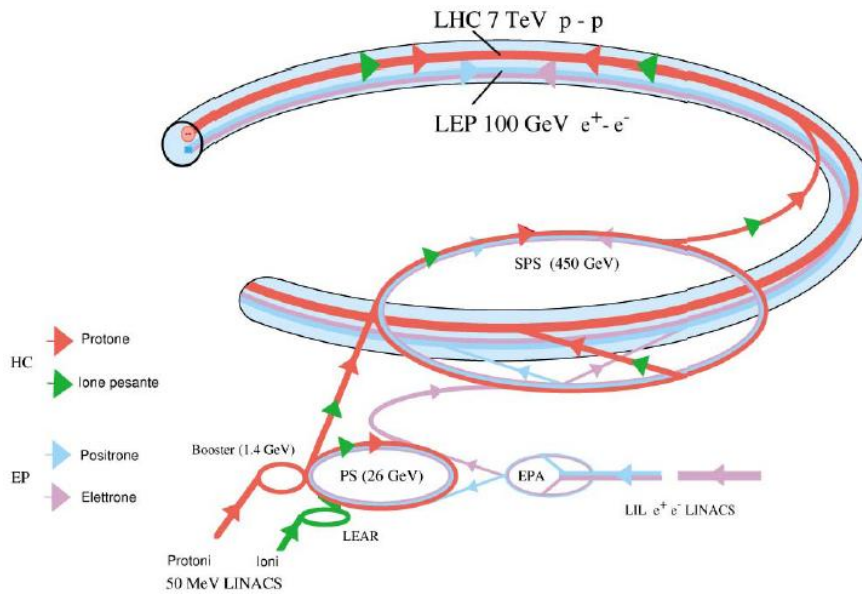


Figure 1.1.: An illustration of the path of the beams injected into LHC (red and green arrows).

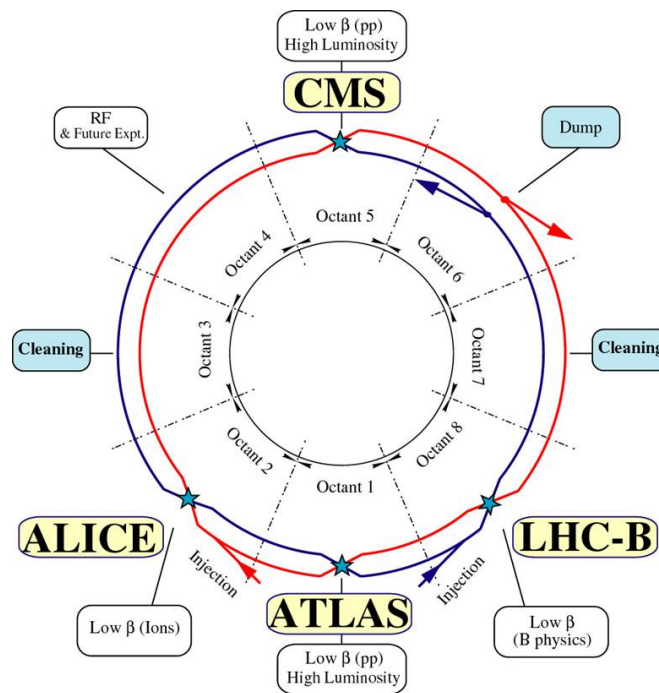


Figure 1.2.: An overview of the experiments in the LHC.

particles. If the Higgs boson is discovered in the LHC, it is discussed that a linear accelerator will be built for further studies.

1.4. Luminosity

While it could be argued that CERN has no real product, all the accelerators and equipment used to create the particle collisions and read-out measurement data are products of CERN. These products are offered to scientists worldwide to enable them to make their particle studies. There are however high demands as a certain beam quality is necessary to achieve optimum conditions for running experiments.

As a measure of the beam quality, the luminosity determines the event rate. The higher this rate, the lower the statistical error of a measured particle parameter. The luminosity \mathcal{L} can be calculated by (see reference [13]):

$$\mathcal{L} = f_{rev} n \frac{N_1 N_2}{4\pi\sigma_H\sigma_V} \quad (1.1)$$

Where f_{rev} is the revolution frequency, n is the number of bunches in one beam, N is the number of particles in each bunch and $\sigma_{H,V}$ are the horizontal and vertical transverse beam dimensions. This shows that the luminosity is inversely proportional to the beam size at the collision points. Since the beam size can not be measured at this location, it is measured elsewhere and then the emittance of the beam is calculated. The emittance is constant around the ring and allows to relate a beam size measurement done at any location of the ring to the beam size at the interaction point.

Therefore the transverse beam size σ is measured by use of a wire scanner, which will be explained in section 1.5, in both the horizontal and vertical dimension at LHC point 4. This allows the emittance ϵ to be calculated as follows (see reference [12] page 225):

$$\epsilon = \frac{\sigma^2}{\beta} \quad (1.2)$$

Where β is a function given by the beam optics of the accelerator. By using the relation between the the transverse beam size and the emittance, which is illustrated in figure 1.5, the beam size at the interaction point can be found and the resulting luminosity can be predicted.

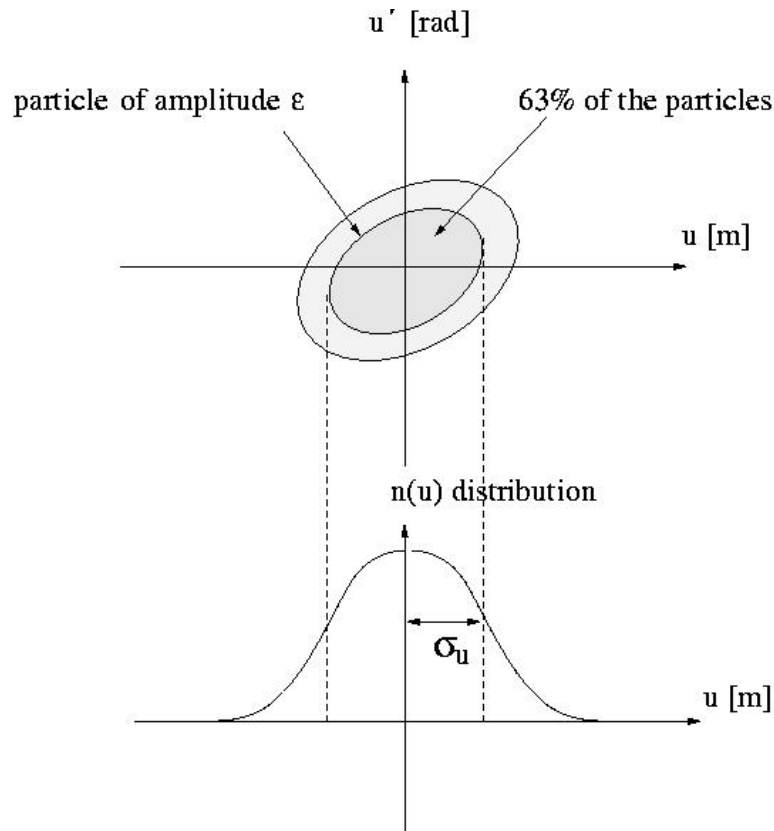


Figure 1.3.: Example of a: Top; Phase space diagram of the beam intensity distribution. The axis are defined as particle position μ and particle angle with respect to the nominal trajectory. Bottom; Projection of the intensity distribution onto the position coordinate.

1.5. Wire scanners

The wire scanner is a measurement instrument used to scatter particles, which creates secondary particle showers and hereby allows accurate measurements of the transverse beam size σ . Due to the great precision, it is used for the calibration of the BGI¹⁰ and BSR¹¹. Hence the wire scanner is more accurate, but the disadvantage is that it only does single scans. The BGI and BSR monitor however the beam continuously, but their accuracy is not that high.

The reason for not using the wire scanner at full intensity in the LHC is firstly because it has been shown (see reference [6]) that the wire-temperature limit will exceed the sublimation point and break the wire. The wire scanner can therefore

¹⁰Beam Gas Ionization

¹¹Beam Synchrotron Radiation

only operate in the LHC when it is partially filled. Secondly, many secondary particles initiated by the proton-wire interaction will be lost downstream, and thus endanger machine components.

Physically, a wire scanner consists of a thin ($\phi 5 - 30\mu m$) carbon wire attached to a fork arm, which is driven by a servomotor (see the principle of a flying wire scanner in figure 1.4).

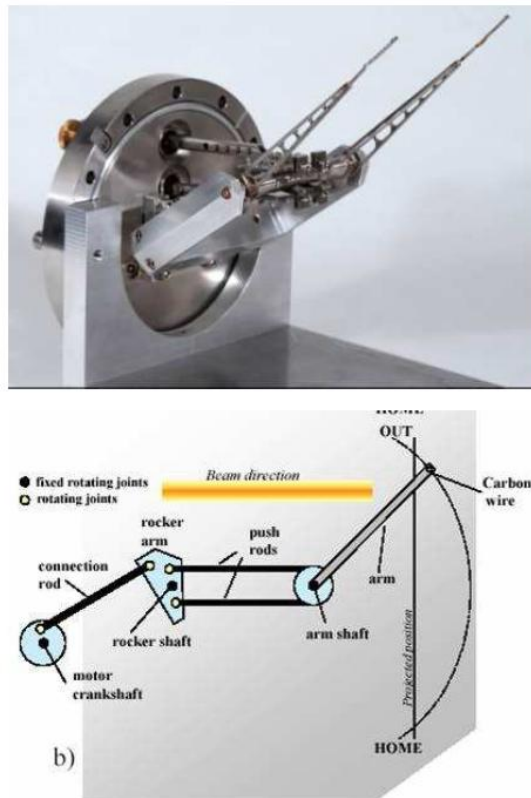


Figure 1.4.: Photo and schematic of operation of a flying wire scanner.

There are two methods of measuring the number of particles impacting the wire. The first principle of the measurements is that an energetic particle beam passes through the thin carbon wire and induces a secondary emission current. Secondly, the beam particles interacting with the wire material and cause a secondary particle shower proportional to the local beam intensity.

In the first method the SEM¹²-signal is measured and amplified. This current can be measured by placing an ampere-meter between the wire and the wall of the

¹²Secondary Emission current

vacuum chamber. By using an proton-to-electron emission coefficient ξ ($\approx 5\%$ for flat surfaces), we can calculate the number of protons interacting with the wire using the measured current in formation.

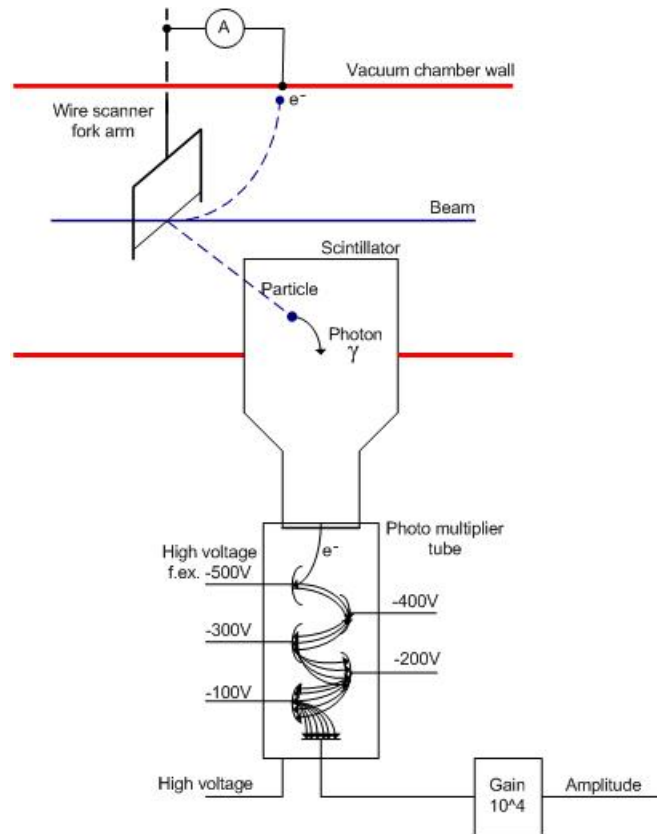


Figure 1.5.: The profile measurement setup using a wire scanner. The scintillator and the photo multiplier tube detects the charged particles of the shower, and the ampere-meter measures the SEM signal.

The other method is to measure the number of scattered particles, which is proportional to the actual number of particles hitting the very thin carbon wire scanning through the beam at a high speed. The scattered particles can be captured with a scintillator, which then transforms the signal and emits it as light to a PMT¹³. The scintillator must be placed at a certain distance and angle from the interaction point, according to where the secondary particle shower will end up. Secondary particles within a certain angle will be absorbed by the scintillator and the ionization produced by the charged particles generates the optical photons, which are re-emitted through a Plexiglas light guide to the PMT.

¹³Photo Multiplier Tube

The PMT is a 6/8/10-stage photocathode tube, which converts the photons to electrons and amplifies the signal. It has a good linearity and high peak current output. The PMT-gains range from 10^4 to 10^6 at the maximum with an overall voltage of 2,5kV.

By acquiring the amplitude of either signal and the position of the wire, the transverse beam intensity profile can be drawn.

1.5.1. How to control the wire scanners

In order to set the various control modes and to trigger wire scanner scans, a control unit is needed to interface it with a remote server. This control unit should be accessible through the VME-bus, which is widely used at CERN. By interfacing it over the VME-bus, control mode registers and scan triggers can be transferred from a remote location, and acquired data can be read out after performed scans.

A FPGA system has been chosen to be the control unit in this project. In the FPGA, there are dedicated memory blocks where various motion control functions can be stored and many processes can interact simultaneously. When using processes synchronized by a system clock, full timing control of the system can be achieved.

To control the wire scanner movement, a scan trigger initiates a function generation clock. This clocks an address counter which steps through a memory containing the motion control data functions. The function data will be converted by a DAC¹⁴ to obtain an analogue control signal. This signal is sent to a motor driver with position feedback, which controls the wire scanner motor and hence the motion of the wire scanner. Meanwhile the feedback position data can be acquired and stored in a memory by a different process.

¹⁴Digital-to-Analog Converter

2. Overview of the system

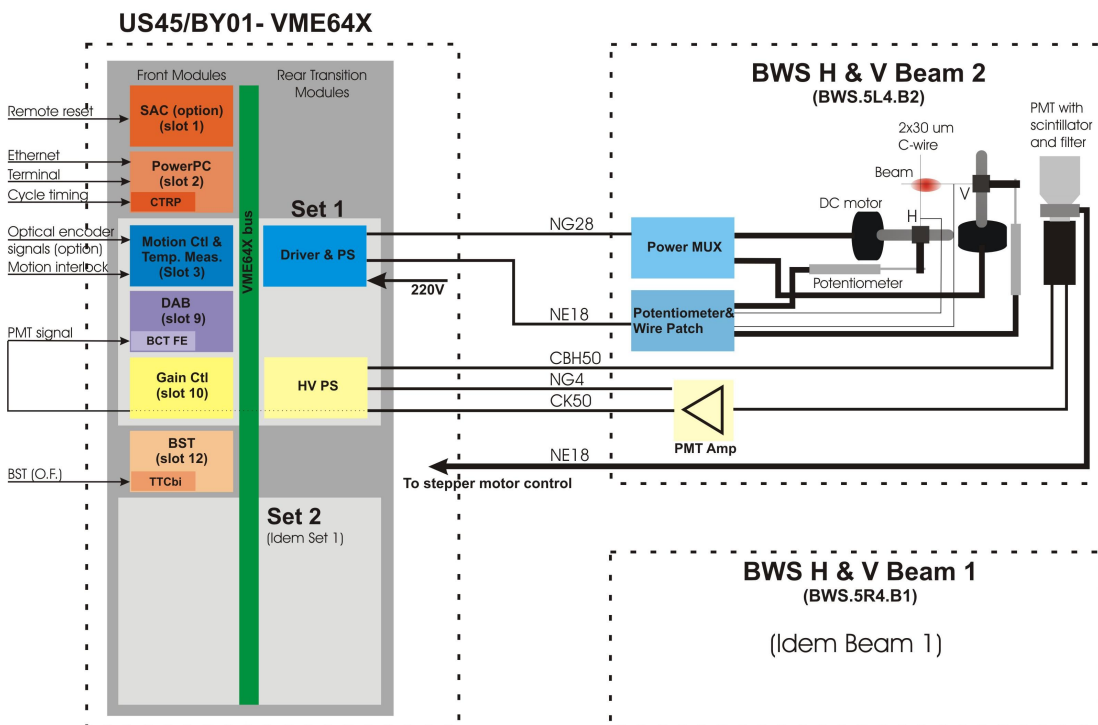


Figure 2.1.: Entire wire scanner system.

As shown in figure 2.1, the LHC BWS system consists of three connected subsystems. The VME64X rack, containing the control cards, is located in an underground area close to the LHC point 4. The WS interaction point for beam 2 (counter-clockwise direction) is located in the 5th cell to the left of LHC point 4, and the WS interaction point for beam 1 (clockwise direction) is located in the 5th cell to the right of LHC point 4. This will be the one single wire scanner system installed in the LHC. However wire scanners are also used in the existing accelerators (PS Booster, PS, SPS). They are of different types and the electronics should also control these.

2.1. PowerPC

The PowerPC is a PC and a protocol bridge, which makes it possible to access the VME-bus (see figure 2.1, green line) through the Ethernet or using a RS-232 terminal. When transferring to or from the PowerPC via Ethernet, the data will be translated from one protocol to the other, IPX / TCP-IP to 2eVME or vice versa.

By using the Ethernet, communication can be established by using a remote computer. When using a terminal, a terminal program accesses the VMEbus through a RS-232 interface and results can be printed out on a screen on site. This is very helpful for testing and debugging the system while on site.

In addition the bus arbitration is included, and the accelerator's cycle timing information is fed to the PowerPC.

2.2. Wire scanner motion control card

The WSMCC¹ chooses which wire scanner to use, controls the motion of the wire scanner, acquires the exact position of the wire and measures the wire temperature. The acquired values must be stored and accessible through the VMEbus interface, this is needed to read out the data and store it on a computer by using appropriate software. There are also additional features like calibrating the potentiometer by using an optical ruler, surveying various signal levels of other cards and acquiring the SEM-signal on the wire. The WSMCC is the main subject of this paper.

2.3. Driver and power supply

The DC motor driver receives the PID² regulated control voltage from the WSMCC and drives the wire scanner.

The card for the motor driver and power supply is mounted on the backplane of the VME64x crate, and it uses only one VME-slot, but it occupies four slots due to its size. This unit mainly contains six 63V capacitors totalizing 126 mF

¹Wire Scanner Motion Control Card

²Proportional-Integration-Differentiation

and power amplifiers, which are all mounted in an EMI³ protected aluminum chassis. The large capacitors are used to feed the wire scanner quickly when it is active. When switching the motor driver off, the time constant of discharge is $\tau_d = R \cdot C = \frac{82}{3} \cdot 126 \cdot 10^{-3} = 3.44s$, then the voltage has dropped to a safe voltage of 24V.

There are three connectors on the back of the driver and power supply chassis. The NG28 cable (see figure 2.1, middle) includes the DC motor voltage and the control signals for the Power MUX. In total, up to four wire scanners can be controlled through this connector. The two NE18 cables (see figure 2.1, middle), which are isolated from the high power signals, contain the signals from the four potentiometers and the low-signaled wire-patch. The signals in the NE18 will be connected to the J2 connector of the VMEbus and each can control two channels. This means that if four wire scanners are installed at one point, both NE18 cables will be used.

2.4. High Voltage power supply and gain control

The HV PS has to supply the PMTs with the high voltages needed to amplify the captured secondary particle signal. These high voltages will be fed through a CBH50 transport cable (see figure 2.1, middle). The NG4 transport cable (see figure 2.1, middle) contains power lines for a remote amplifier and PMT gain control signals. The PMT gain control signals come from the gain control card and are fed through the HV PS card. The CK50 transport cable (see figure 2.1, middle) returns the real intensity amplitude, feeds it to the DAB card through both the HV PS card and the gain control card.

2.5. Digital acquisition board

Synchronized by the bunch timing, the DAB card stores the intensity data it receives from an acquisition ADC in memories. This has to be configured to synchronize to the same bunch number as in the WSMCC. These memories can then be read out through the VME interface.

³Electromagnetic Magnetic Interference

2.6. BST - Beam Synchronization Timing

The beam synchronization timing card has two receivers, which receive information of the timing for the two opposite directed beams. The timing for each bunch can be obtained and the BST card creates two 8-bit signals, which resemble the active bunch number of each beam. The card is inserted in the middle VME-slot. It sends timing data for one beam to one side of the bus, and the timing data for the opposite directed beam to the other side of the bus.

3. Project planning

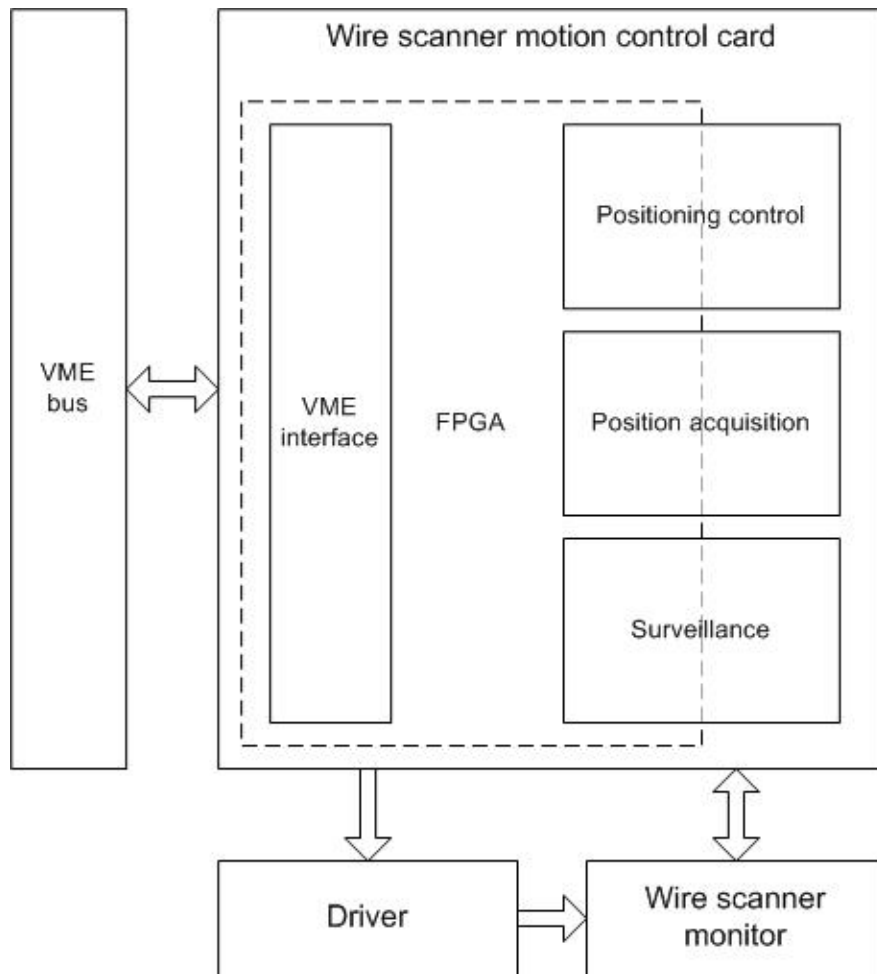


Figure 3.1.: An overview of the wire scanner motion control card. A complete functional diagram can be found in appendix A.

3.1. Wire scanner position control

In order to use various wire scanners in different types of accelerators with different types of beams, the speed of the scan and acquisition rate must be adjustable. The idea for solving this issue is to use an adjustable clock divider, which clocks an address counter pointing to a memory. The memory is expected to contain $2^{12} = 4096$ words with a width of 12 bits (motor voltage resolution of 4096 levels), which represents the function of the wire scanner movement.

At first, the plan was to use a SRAM¹ for storing the digital position values of the motor current function, since various test patterns will be loaded. A second opinion led however to an internal ROM² of the FPGA, as the size of the required memory is quite small, so having an extra external SRAM or flash memory is not necessary and causes more timing and reliability issues. The internal ROM can be loaded with a memory initialization file through the JTAG³-interface.

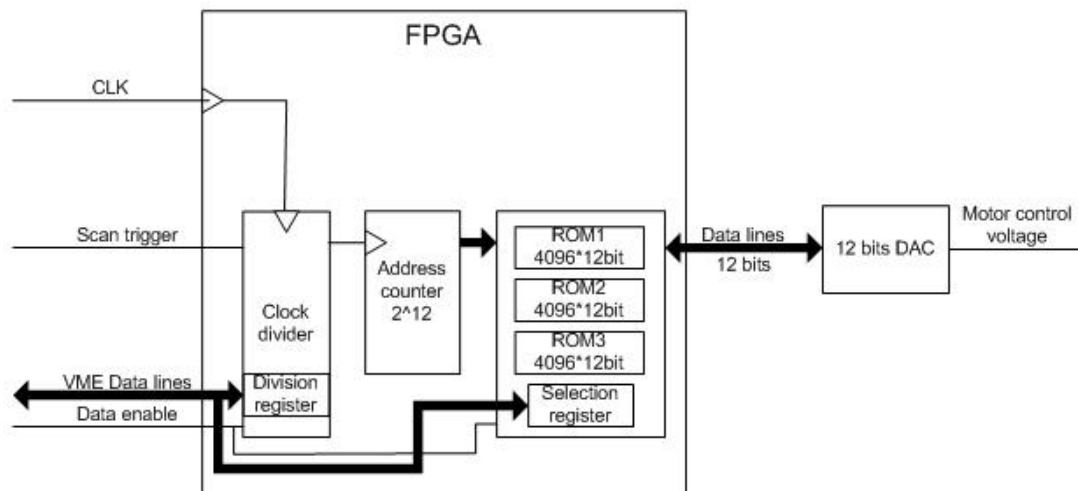


Figure 3.2.: Wire scanner position control.

The motion controller will wait for a trigger to start the scan, and the motion will be controlled by clocking through the memory values at a predefined speed. By stepping through the addresses and converting the data by a DAC, the amplitude of the analogue DAC-output signal will be compared to a motion feedback signal by means of a PID regulator and output a control voltage for the

¹Static Random Accessible Memory

²Read Only Memory

³Joint Test Action Group

wire scanner motion. This control voltage is amplified by the motor driver, which is a high power H-bridge motor driver.

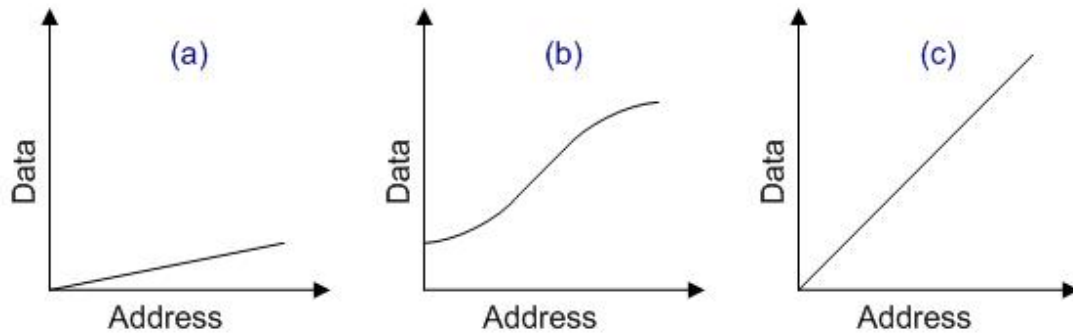


Figure 3.3.: Wire scanner movement functions: a) Linear offset mode; b) Accelerated and decelerated fast scan mode; c) Linear slow scan mode. The exact profiles will be calculated in section 4.5.2

As shown in figure 3.2, there are three ROMs. These will be fed with different functions (see figure 3.3), and a register selects which function should be used during a scan. When using the fast scan mode, the linear offset profile must first be used to bring the wire scanner away from the hard-end. When the wire is in the offset position, the address can be cleared and the fast scan mode activated. The same procedure must be performed when bringing the wire to the hard-end home position, only then the address must be set before the linear offset mode brings the wire home. More details follow.

The linear offset mode (see figure 3.3a) is used to prevent the wire scanner to hit to the hard-end position for each scan in fast scan mode. With its overshoot, this would damage the wire-scanner severely. The offset position is currently set to be at around 15% of the stroke.

The fast scan mode (see figure 3.3b) has an acceleration part starting at the given offset, a constant speed in the middle part (where the beam is located) and deceleration in the end part (stopping the wire scanner by the upper offset). This mode should be used during hi-speed scans.

The linear slow scan mode (see figure 3.3c) is simply a profile with constant speed. This works however only when the speed is low enough for the motor to follow, and can thus be taken all the way to the hard-end position. The slow scans are performed during beam tail-detections.

3.2. Acquisition

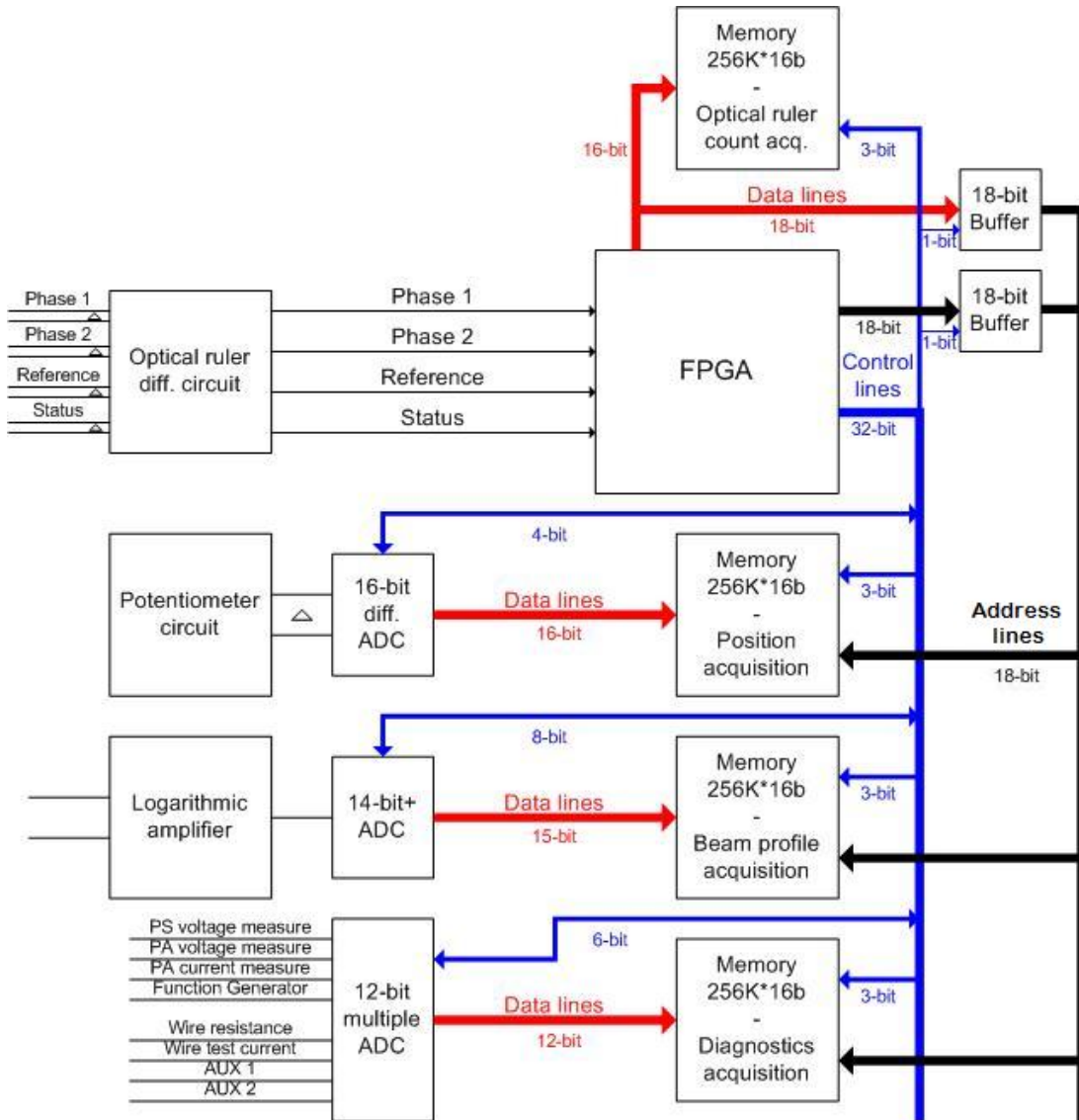


Figure 3.4.: Wire scanner position acquisition diagram.

3.2.1. Wire scanner position acquisition

During the wire scanner movement, a voltage difference will be detected by comparing the DAC output voltage with a measured position voltage. This means that there is a slight delay from comparison to difference compensation, which results in a movement which is not exactly the same as the DAC voltage output.

In order to maintain the precision of the profile measurements, the actual position data must be acquired.

For measuring the wire position, a potentiometer has been attached to the fork-arm of the wire scanner. The position will be acquired by measuring the voltage at the potentiometer. This voltage will be digitized with a 16 bit differential ADC and stored in a SRAM (see figure 3.4). The ADC is using a SAR⁴ for the data conversion, which is testing if the voltage is lower (gives 0 at current bit) or higher (gives 1 at current bit) than the current bit voltage representation, starting on the MSB⁵ and ending on the LSB⁶. As this is a 16bit ADC, the conversion then takes at least 16 clock cycles for the conversion. The conversion start will be triggered by the acquisition clock, and the SRAM write signal must be delayed by the conversion time of the ADC.

However, as the potentiometer is not fully linear, it must be calibrated before it is implemented in the final application. For the calibration, a stable linear optical ruler will be used to create a calibration-LUT⁷ with a resolution of $1\mu m$. This LUT will be stored in the PowerPC software of the crate, which will compare and correct the potentiometer values in accordance to the LUT in the final application.

Optical ruler operation

The optical ruler has three differential outputs, which will feed a Quad Differential Line Receiver through digitizing electronics. The three logical outputs from the differential line receiver then consist of two incremental position signals (see figure 3.6) with a phase difference of 90 degrees and a mid-position reference signal.

The optical ruler has two mechanical parts, a movable scanning head and a fixed ruler. A light is emitted by a LED⁸ mounted in the movable scanning head. The light emits through four grated scanning windows and is reflected by the ruler. An incremental track graved in the ruler causes periodic variations in the light reflection, which is registered by four solar cells corresponding to the scanning windows. The solar cells are respectively phase-shifted by 90 degrees, and by means of the scanned light variations two electrical sine waves are derived.

⁴Successive-Approximation Register

⁵Most Significant Bit

⁶Least Significant Bit

⁷Look-Up-Table

⁸Light Emitting Diode

The second sine-wave is 90 degrees phase shifted with respect to the first sine-wave in order to double the counts and indicate the movement direction. Also two additional scanning windows are available to scan a reference track and create a peak signal at the reference point.

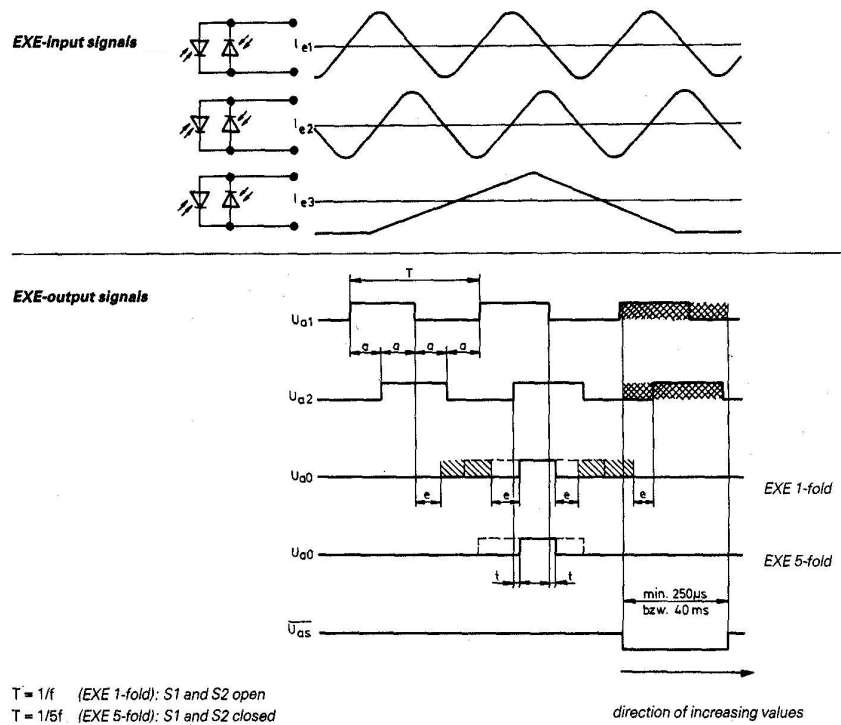
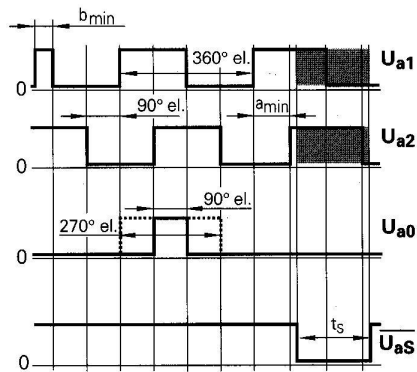


Figure 3.5.: Optical Ruler sine-wave signals fed to and digitized by the EXE 610C.

As shown in figure 3.5, the sine waves and the reference signal induced by the optical ruler are fed onto a digitizing circuit. The drawn digital output signals have a 1-fold interpolation. This means that the only interpolation point is at ground, and the digital output gives a high signal for the positive half-period and a low signal for the negative half-period. With a grating pitch G_p of $40\mu m$, every change of phase state indicates a move of $10\mu m$, as there are four different phase-states in the two pulse trains. The currently used EXE 610C digitizing circuit supports however 5- and 10-folded interpolation, which results in a resolution of $2\mu m$ and $1\mu m$ respectively.

In order to achieve the $1\mu m$ resolution r , a 10-folded interpolation is therefore needed. As the EXE 610C digitization circuit has a clock frequency of $f_t = 2MHz$, the maximum input frequency is $f_i = \frac{f_t \cdot r}{G_p} = \frac{2MHz \cdot 1\mu m}{40\mu m} = 50kHz$. A

maximum input frequency of 50 kHz results in a maximum detectable movement speed of $v_{max} = \frac{G_p}{T_i} = \frac{40\mu m}{1/50kHz} = 2m/s$.



Incremental signals: Square-wave pulse trains U_{a1} and U_{a2} with 90 el. phase shift, plus their inverted pulse trains $\overline{U_{a1}}$ and $\overline{U_{a2}}$ (according to RS-422).

Min. edge separation a_{min}

Min. pulse width b_{min}

Reference pulse: Square-wave pulse U_{a0} and its inverted pulse $\overline{U_{a0}}$

Width 90° el. (standard);
switchable to 270° el.

Fault detection signal: Square-wave pulse $\overline{U_{aS}}$

Duration $t_s \geq 20$ ms

Max. response duration after switch-on: 0.5 s.

With $U_{aS} = \text{low}$ (tristate), the outputs U_{a1} , U_{a2} and $\overline{U_{a1}}$, $\overline{U_{a2}}$ can be switched to high impedance. Standard setting: tristate inactive.

Figure 3.6.: Digitization circuit phase train signals.

The EXE 610C has a clock frequency f_t of 2 MHz, which causes the minimum edge separation and pulse width (see figure 3.6) to equal one clock cycle of $0.5\mu s$.

The differential outputs from the digitization device will be transformed to single-ended signals and fed to the FPGA. In the FPGA, a counter will count either up or down, depending on the phase-shift between the two incremental signals. A count will occur every time when the state of phase changes. For every count, the wire scanner has moved $1\mu m$ (by a 10-fold interpolation setting), hence a 130 mm motion (one scan length) results in 130000 counts.

One might wonder why the optical ruler cannot be used in the final LHC application. The reason it cannot be used is due to the high radiation in the LHC, which will damage the optics of the optical ruler and reduce its lifetime drastically. This could then cause phase errors, resulting in incorrect position data. The optical ruler can however be used in the SPS ring. In the SPS ring, the required accuracy is $4\mu m$, so the two lower bits can be left out while the upper 16 bits are stored into a SRAM (see figure 3.4).

ADC, digitization of the potentiometer voltage

The potentiometer has a differential voltage read out, which cancels interference due to environment better than a single ended voltage and thus the accuracy is well improved. To digitize the differential voltage, a differential ADC is needed.

The chosen differential ADC in this application is the AD7677, which is asynchronous and has a 16-bit resolution. The AD7677 uses a SAR, which uses a track/hold circuit for holding and sampling the signal triggered by the conversion signal. With a maximum throughput of 1 MBPS, this ADC is ideal for precise data acquisition systems, due to both fast sample rate and the accuracy.

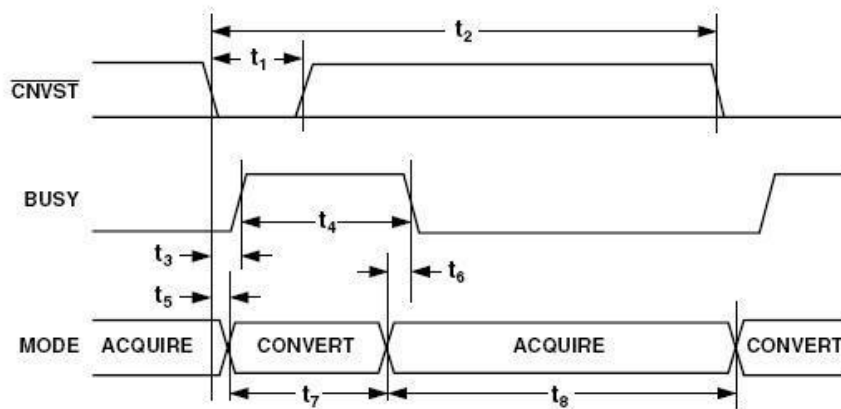


Figure 3.7.: Basic timing diagram for the AD7677.

As seen in figure 3.7, the data digitization is triggered by the falling edge of the conversion start signal. The busy signal goes high to indicate that the ADC is busy sampling the signal held by the hold-and-track circuit. When the sampling finishes, the busy signal is released and the data acquisition can commence.

3.2.2. PMT and SEM acquisition

The signal is normally acquired by a DAB card. A logarithmic amplifier was selected for PMT tail studies, but in the PSB, the SEM signal (see section 1.5 for detailed information) is normally connected to the logarithmic amplifier. The output signal from the logarithmic amplifier will be digitized by a 14-bit ADC and can be stored in a SRAM (see figure 3.4). This ADC also uses a SAR for the data conversion, and the SRAM write signal must be delayed by the conversion time.

As the logarithmic amplifier flattens the beam profile, it is useful for tail studies of the beam. During beam tail studies, it is preferable to use the PMT signal which is then connected to the logarithmic amplifier. The PMT signal has a longer amplitude than the SEM signal. In the PS and SPS rings, the PMT might be connected to the logarithmic amplifier by default due to lower speed requirements, and the high speed DAB card may be left out of the system.

ADC, digitization of the logarithmic amplified voltage

The logarithmic amplified output signal is a single ended voltage, and the resolution requirements are not as high as for the potentiometer.

The selected ADC is the AD7484, a 14-bit ADC using SAR to avoid pipeline delays. The AD7484 offers a 15th bit, which can be used to indicate an over-range up to 8% of the nominal range. Also there is an offset register, to which a 12-bit value can be written in a twos complement format. This offset value ranges from -1310 to +1310 which represents an offset of -200 mV to +200 mV. This value can be written to the ADC by writing a value to a VME address which is linked to the offset register of the ADC.

There are two different parallel modes, and the chosen mode is the parallel mode 1. By using parallel mode 1, the current data is valid at the rising edge of the busy signal (as shown in figure 3.8) and not as the previous data by the next conversion start. When using this mode, the same VHDL-component can be used for both the AD7677 and AD7484, only one busy signal must be inverted.

3.2.3. Diagnostics acquisitions

There are many circuits involved during a scan which are influenced by their environment as they are not ideal. To check these signals during a scan, the respective voltages have been connected to a multiple input ADC. This ADC can be programmed for which channel it should use, or set sequencers to sequence through all channels or a set of selected channels.

By measuring the Power Supply- or Power Amplifier-voltage during a scan, diagnostics for a unloaded- and a loaded-circuit can be compared for troubleshooting and statistics. The diagnosis for the wire resistance and the thermo ionic current makes it possible to find out how much heat is induced on the wire

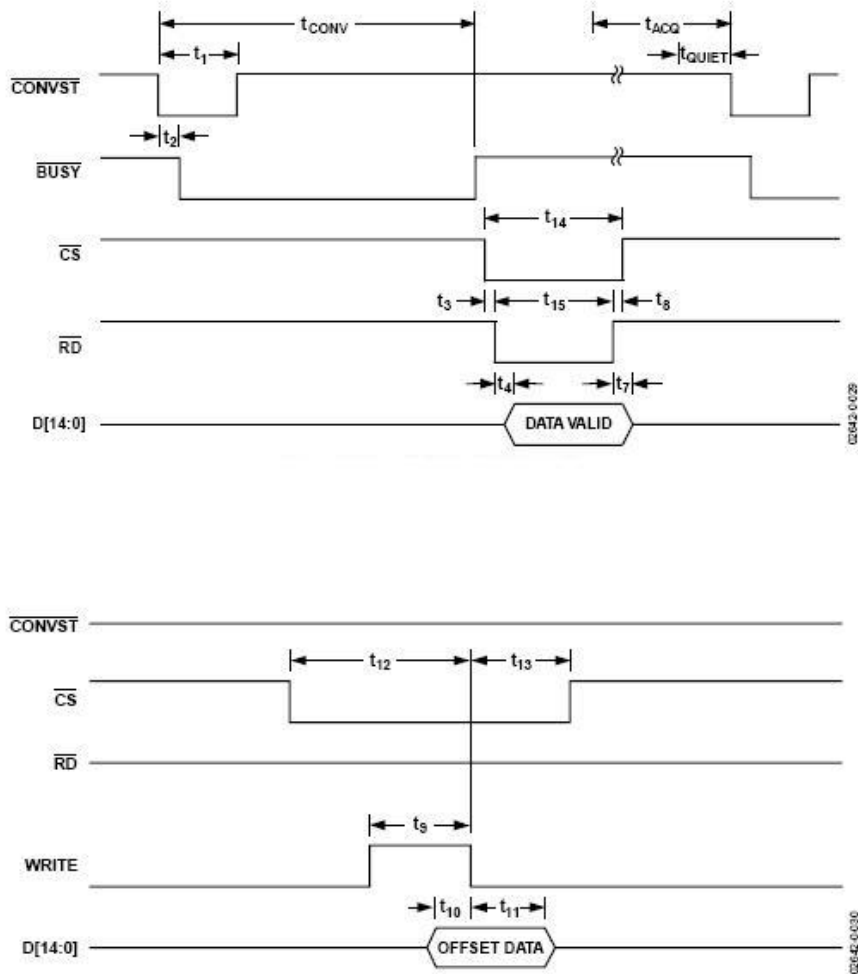


Figure 3.8.: Basic timing diagram for the AD7484: Top; Read cycle timing. Bottom; Write cycle timing.

at whichever stage of the scan. This will have to be calculated by the wire's temperature coefficient, and thus solved in the PowerPC crate software. The input signal fed from the function generator output voltage is used to check that the output function for the motion control does not contain any glitches. In addition, there are two supplementary auxiliary inputs, where the user may connect signals he finds worth acquiring, like a laser calibration signal.

ADC, digitization of the diagnostic voltages

For the diagnostic signals, there are many channels which are not used every time. Thus a programmable multiplexed ADC is needed, where all the sequenced output data will be stored in the same SRAM. The organization of the SRAM would then depend on the sequence chosen in the ADC-multiplexer, but this can easily be solved in the PowerPC crate software.

For this purpose, the AD7938 has been chosen. This is an 8-channel, 12-bit ADC with a programmable sequencer. It has a maximum throughput of 625 kSps and uses a SAR with track/hold.

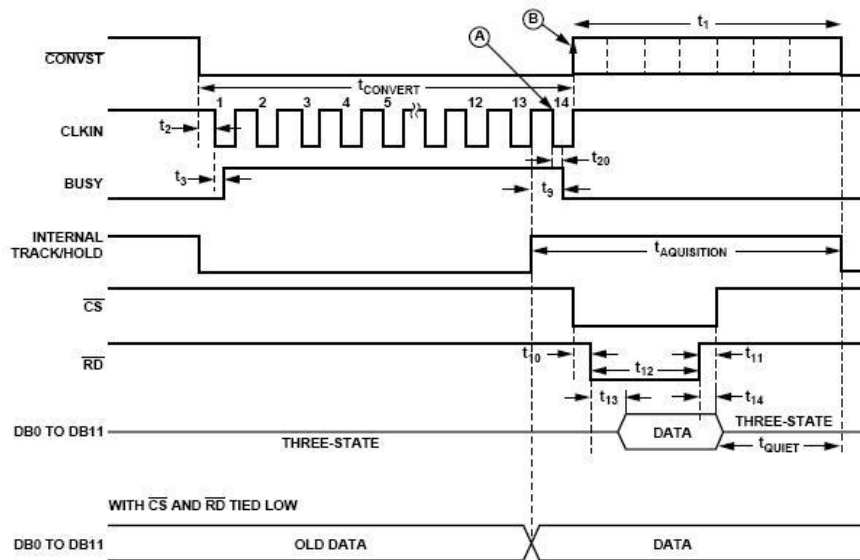


Figure 3.9.: Basic timing diagram for the AD7938 in parallel word read mode.

As shown in figure 3.9, the conversion principle is similar to the AD7677 and AD7484, but the conversion start should be held low during the entire conversion and an external conversion clock is needed.

To select which channels to convert and conversion operation, there are two write-only registers. A 12-bit control register defines the operation mode and if therein selected, an 8-bit shadow register defines which channels should be selected.

The ADD[2..0] bits are address bits and defines either which channel to convert or which is the last channel that should be converted in a sequence starting on the least significant channel. The operation is chosen depending on the sequence

DB11	DB10	DB9	DB8	DB7	DB6
PM1	PM0	CODING	REF	ADD2	ADD1
DB5	DB4	DB3	DB2	DB1	DB0
ADD0	MODE1	MODE0	SHDW	SEQ	RANGE

Table 3.1.: Control register bits in the AD7938 ADC.

bits set in the SEQ- and SHDW-bits.

The SEQ and SHDW bits work in conjunction and therefore have four possible modes. If both bits are set low, the channel to be converted is selected by the ADD[2..0] bit contents. If the SEQ bit is set to logic low and the SHDW bit is set logic high, the shadow register will be used. The shadow register will then be filled on the next write operation, and the set high bits will be converted consecutively (see table 3.2). If the SEQ bit is set to logic high and the SHDW bit set to logic low, then an un-interruptible sequence is selected, which allows other bits to be changed between conversions. If both the SEQ and SHDW bits are set to logic high, then a consecutive sequence starting from channel 0 through the channel number set in ADD[2..0] is selected.

DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
V_{IN7}	V_{IN6}	V_{IN5}	V_{IN4}	V_{IN3}	V_{IN2}	V_{IN1}	V_{IN0}

Table 3.2.: Shadow register bits in the AD7938 ADC. The bits which are set logic high are active in a sequence operation if the SHDW bit is set high and the SEQ bit is set low in the control register.

For further details on the remaining bits shown in table 3.1, refer to the AD7938 datasheet.

3.2.4. SRAM, Storing acquisition data

There are four SRAMs on board, storing the acquired data. The chosen SRAMs are all the IS61LV25616AL, which is an asynchronous CMOS static RAM and has 256K words of 16 bits. This is enough to acquire a 1 μm resolution stroke, as that will result in 130 000 words (for a 130 mm stroke of the linear wire scanner).

As the memory is asynchronous, no clock or refresh signal is needed and the access time is 10-12 ns. There are mainly two ways to control the SRAM, it can either be address controlled or chip enable controlled. The latter is however a

more secure way to be sure the correct data is loaded to/from the correct address, power consumption is lower and full control of the memory data flow is gained.

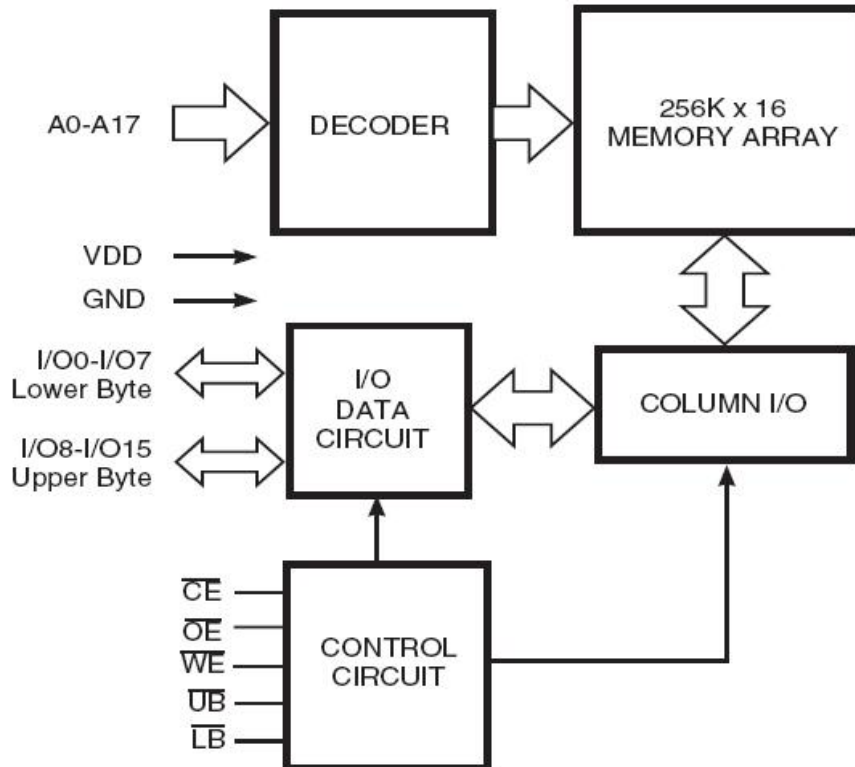


Figure 3.10.: Block diagram of the SRAM.

Both the lower and the upper byte control inputs will be connected to ground, so there will always be transfers of 16 bits. For a read cycle, the active low write signal stays high, the active low output enable will be pulled low and the new address set at the input before the falling edge of the active low chip enable occurs.

For a write cycle, the active low write signal is pulled low, the active low output enable is a don't care and the new address must be set at the input before the falling edge of the active low chip enable occurs.

3.3. Surveillance

To make sure that the wire scanner motion control card is not exposed to errors, which may cause hazardous disfunctions, it is necessary to survey certain values

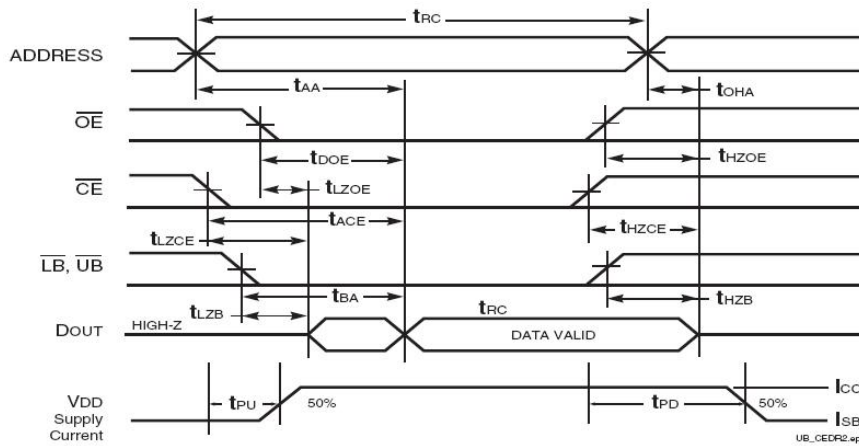


Figure 3.11.: Read cycle, controlled by chip enable.

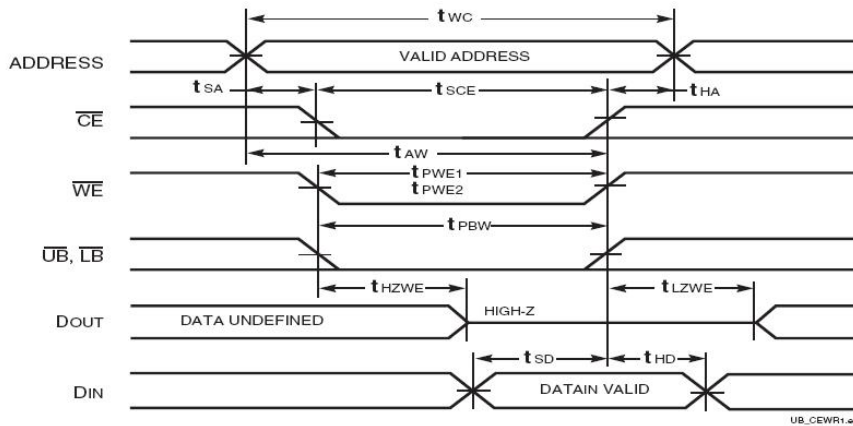


Figure 3.12.: Write cycle, controlled by chip enable.

for error handling. If a surveyed value fails or exceeds a set risk-level, the normal run should halt and the wire brought back home in order to prevent fatal errors from occurring.

Many signals (see appendix C) will be treated to be indicated as either a high or low value, where a high value means no error and a low value means error. The signals will be sampled at a lower speed, say 1000 Hz. These will be registered, and if an error occurs, its corresponding code will be displayed on a 7-segment display. A registered low value will be held until it has been read out and reset through the VMEbus.

3.4. VME interface

To be able to remotely control the hardware on the board, some communication interface must be used, and at CERN the VMEbus is widely used for this purpose.

It supports multiple bus masters and high data transfer rates. The topology which will be used in this project, is the extended VME64 Standard, VME64x. This topology normally uses a 2eVME bus cycle, which doubles VME64x's backplane performance by a two edge handshake for each data transfer. Original VME and VME64 transfers use a four edge handshake for each data transfer.

A VME64x-bus chassis consists of a card cage with 2 - 21 slots and a backplane with three connectors. Slot 1 is the System Controller slot, but this can be overruled. Cards with different functions are inserted in the slots to form a customized VMEbus chassis, which makes the VME controlled system (see the wire scanner system in figure 2.1). Most VMEbus cards are configured via a combination of hardware jumpers, card-specific software configuration, and setting parameters in a non-volatile memory.

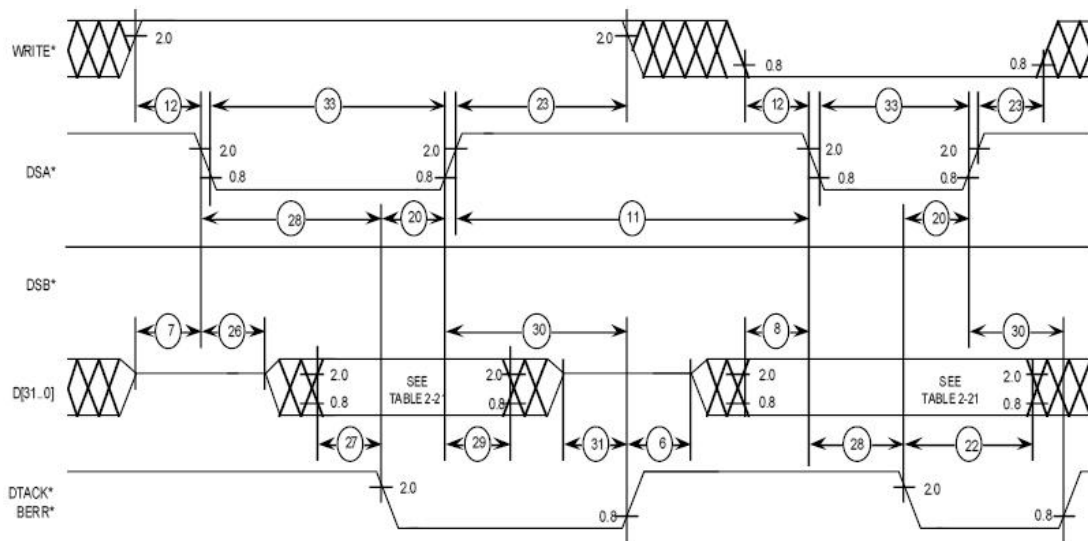


Figure 3.13.: VME-bus timing diagram. The edges of DSx depend on the AS edges.

This project does not demand high transfer speed and a simple VME interface is to be made, where the data acknowledgement is set to a certain delay-count long enough to process the data. Due to pin restrictions on the flat pack FPGAs, a minimum of the VMEbus-signals have been selected for basic VME operation.

This means the needed signals are basically the 24-bit address lines, 6-bit address modifier lines, the address strobe, 16-bit data lines, 2-bit data strobes, the write signal and the system reset signal.

A falling edge of the address strobe indicates that the addressing related lines are ready. Hence by this falling edge, the address lines, the address modifier lines and the write signal can be registered. By the falling edge of the data strobes, the data is ready to be sent or received. The actual transfer will occur when the slave pulls the data acknowledge signal low, and after the transfer has been fulfilled, the address and data strobes are released (set to inactive high state).

4. FPGA design

4.1. General about FPGAs and VHDL

An FPGA is an IC¹, which consists of thousands of logic cells or modules. These are structured so they can be interconnected by a matrix of wires and programmable switches, determining which gates are connected where.

VHDL² was developed as the standard language for describing the structure and function of ICs, and to unite the various VHSIC³ contractors.

VHDL is well suited for designing processes, because it has an architectural structure, where a system is composed of subsystems and these are interconnected. All the system functions are described by standardized codes, which are similar to other program languages. The abstract code environment is then synthesized into detailed structures, and the functionality can be verified in a simulator.

The synthesis is the compiler of the VHDL code, hence by synthesizing, the VHDL code is translated into a net list forming a schematic of gates and flip-flops. There are three different synthesis classes: logic, RTL⁴ and behavioural synthesis. The logic synthesis minimizes Boolean expressions into gates. The RTL synthesis does the same as the logic synthesis, but in addition it also translates sequential statements into gates and flip-flops. The behavioural synthesis reuses hardware components in several parallel language constructions.

¹Integrated Circuit

²Very-High-Speed-Integrated-Circuits Hardware Description Language

³Very High Speed Integrated Circuits

⁴Register Transfer Level

4.2. FPGA selection

It is desired to use Altera chips, due to broad support both online and by other users at CERN. Besides the radiation requirements are not that high, as the WSMCC will be installed in an isolated area aside the tunnel.

It has been decided that the final board shall contain three FP⁵ (PQFP⁶ or TQFP⁷) FPGAs. The main reason for selecting three FP FPGAs instead of one BGA⁸ FPGA, is due to the routing difficulties when using the BGA package, where many layers are needed. Also the routing becomes less centered to one component, the various functions may easier be divided into own groups on the card layout and it is easier to access the pins with a scope. This means that each separate FPGA will deal with different functions. The master FPGA will decode the VME interface, control the dataflow on the card and activate processes of the slave FPGAs. One slave FPGA will contain the optical ruler calibration/acquisition and the function generator, while the second slave FPGA will survey and display occurred error warnings.

This project demands neither DSP processing, nor advanced PLL functions, nor the Nios II soft-core processor (which demands quite some memory blocks). This is why the three chosen FPGAs are from the Cyclone II family (see table 4.1 for specific package-data), which is a 2nd generation low-cost FPGA produced by Altera. By choosing FPGAs within the same family, the configuration is simplified (see section 4.3). More detailed information concerning the selection of the package sizes is included in the appropriate sections (Master FPGA 4.4, Slave 1 FPGA 4.5 and Slave 2 FPGA 4.6).

4.3. FPGA configuration

An FPGA normally uses SRAM cells to store configuration data. However, data is lost at power-down as the SRAM is volatile. Therefore the configuration data must be stored in a non-volatile memory and reconfigure the FPGAs at power-up.

The Altera Cyclone II family is supported by a EPCS, which is a serial

⁵Flat Pack

⁶Plastic Quad Flat Pack

⁷Thin Quad Flat Pack

⁸Ball Grid Array

	Device EP2C5Q208C8	Device EP2C5T144C8	Device EP2C8T144C8
FPGA Family	Altera Cyclone II	Altera Cyclone II	Altera Cyclone II
Package type	PQFP	TQFP	TQFP
Speed grade (ns)	-8	-8	-8
Logic Elements (LEs)	4 608	4 608	8 256
RAM bits	119 808	119 808	165 888
M4K RAM blocks	26	26	36
User pins	142	89	85
PLLs	2	2	2
Embedded multipliers	13	13	18
Size (mm x mm)	22 x 22	22 x 22	30.6 x 30.6

Table 4.1.: Data for the chosen Altera Cyclone II FPGAs (see reference [14]) to be used on the control card.

configuration device from Altera. The EPCS is based on Flash memory, and there are four sizes, 1-, 4-, 16-, and 64-Mbit. As the uncompressed Raw Binary File Size (Bits) for an EP2C5x device is 1'265'792 and for a EP2C8x device is 1'983'536 (see table 4-3 in reference [15]), the total RAM bits required for the configuration data is:

$$TotalRAMbits = (2 \cdot 1265792) + (1 \cdot 1983536) = 4515120RAMbits \quad (4.1)$$

As this exceeds 4 Mbits, either the 16 Mbits or the 64 Mbits configuration device must be used. They both have the same package size, but as 16 Mbits is already more than sufficient and costs half of the 64 Mbits package, the 16 Mbits package has been chosen for this project.

4.3.1. Supported programming schemes

The EPCS device can be filled through two different configuration schemes, directly by using a download cable in AS⁹ mode or indirectly through a serial flash loader bridged to the JTAG interface by the master FPGA.

⁹Active Serial

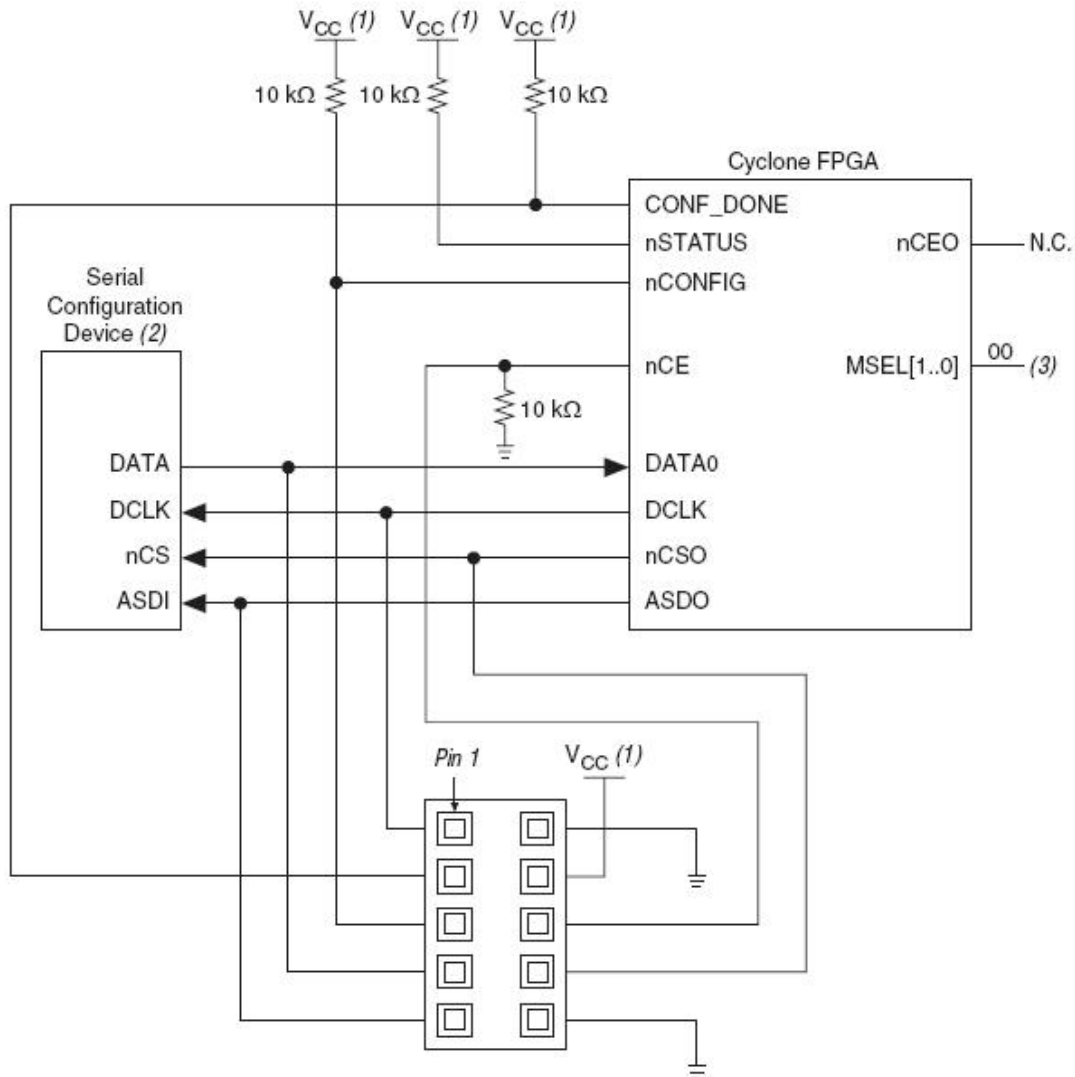


Figure 4.1.: FPGA and Serial configuration device programming in AS mode.
Figure origins from figure 4-2 in reference [15]

By using the direct configuration scheme (see figure 4.1), a download cable accesses the ASMI¹⁰ and can configure both the FPGAs and the serial configuration device in AS mode. AS mode basically means that the FPGA is set as the master, which controls the data flow, and the configuration device is the slave. The alternative configuration mode is the PS¹¹ mode, which latches data from the configuration device to the FPGA, but this mode is not supported

¹⁰Active Serial Memory Interface

¹¹Passive Serial

for the serial configuration device. When using the ASMI interface, only the programming modes can be used, as there is no support for test modes. Also a separate download connector is required, which takes up space on the PCB¹² and the front panel.

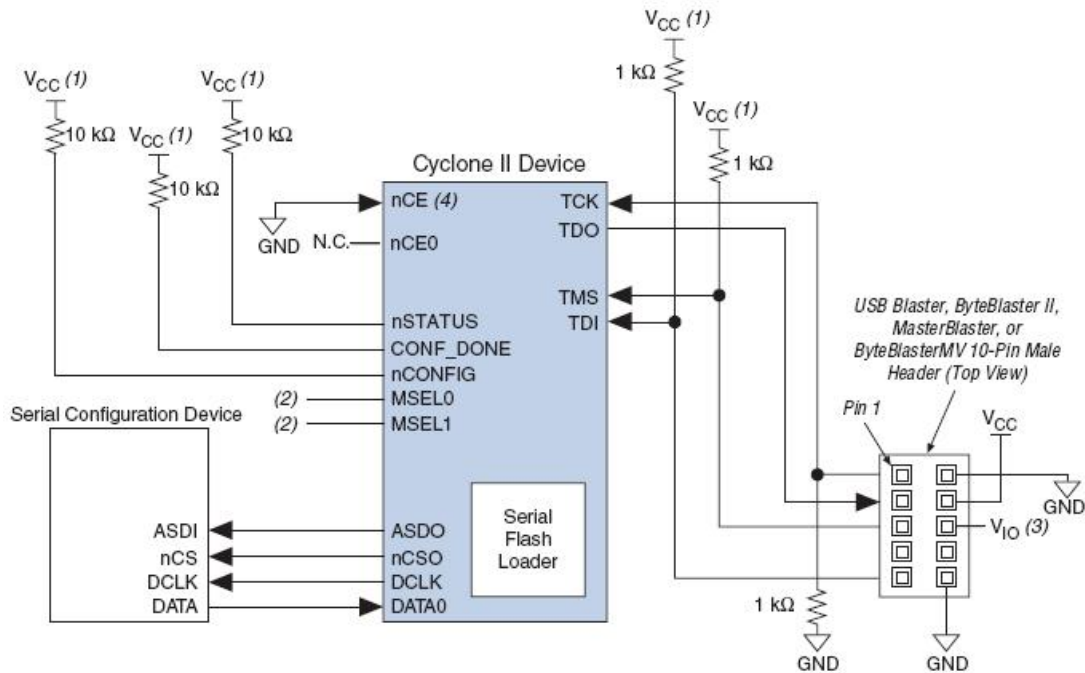


Figure 4.2.: JTAG Configuration of the FPGA and indirect Serial Configuration device programming through the Serial Flash Loader. Figure origins from figure 4-25 in reference [15]

The indirect configuration scheme (see figure 4.2) for the Serial Configuration device is through a JTAG interface. The JTAG interface loads a Serial Flash Loader to the master FPGA, which is a bridge between the JTAG interface and the ASMI interface (see figure 4.3). The master FPGA is set as the master in the AS mode configuration, and controls the dataflow of the ASMI interface. After configuration of the Serial Configuration device, nCONFIG can be pulled low in order to initiate a reconfiguration. The Serial Flash Loader is then removed from the FPGA, and the configuration data will be loaded from the Serial Configuration device. The JTAG interface can also configure each FPGA in the JTAG chain separately, and it supports advanced test functionalities

¹²Printed Circuit Board

like Boundary Scan, In-System Memory Content Editor and Signaltap II Logic Analyzer.

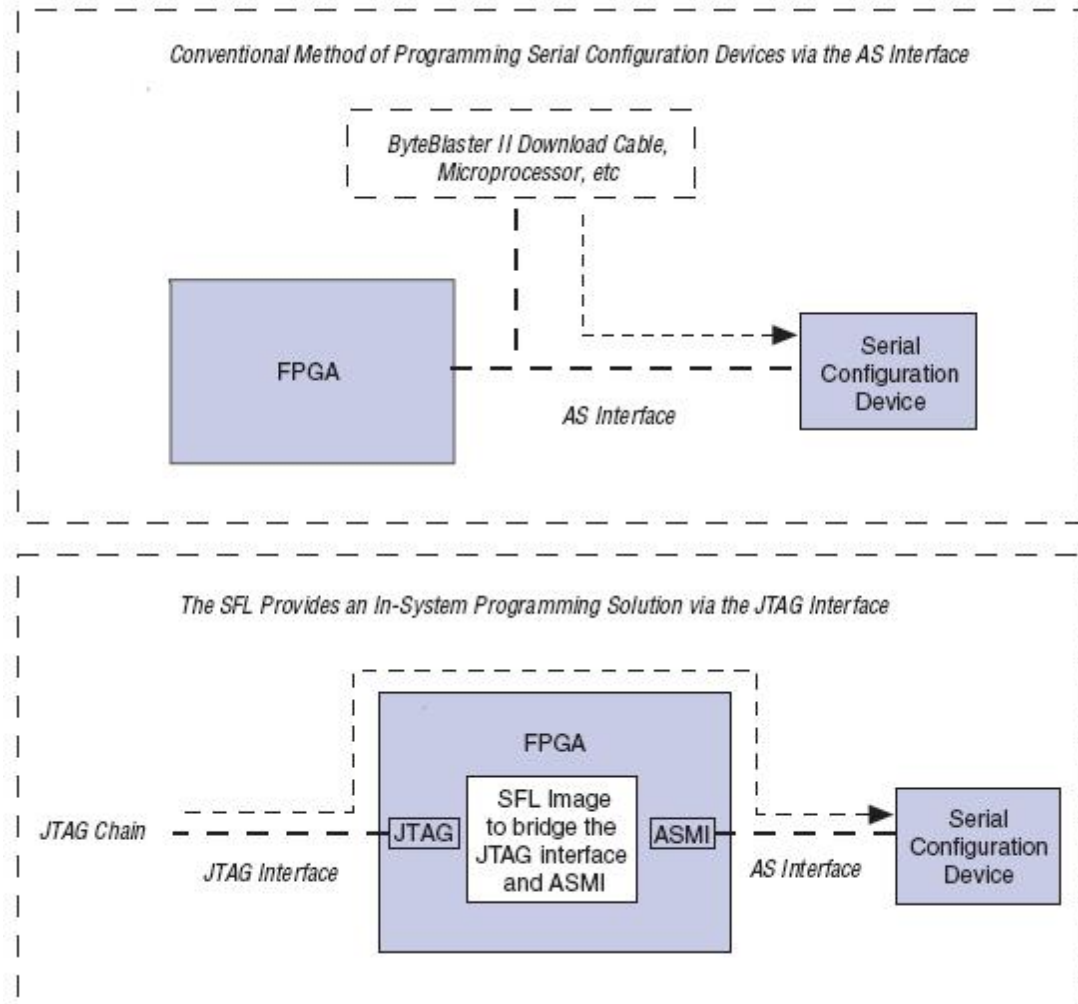


Figure 4.3.: Principle of the Serial Flash Loader interface bridge compared to the conventional configuration scheme. Figure origins from figure 1 in reference [16]

4.3.2. Which configuration scheme to use

The final motion control card is quite dense, so space limitations make the solution using only one connector a better option. This means that one download connector will be connected to the JTAG interface, and there is a combined JTAG and AS programming scheme. As there are multiple FPGAs, a mixed scheme (see

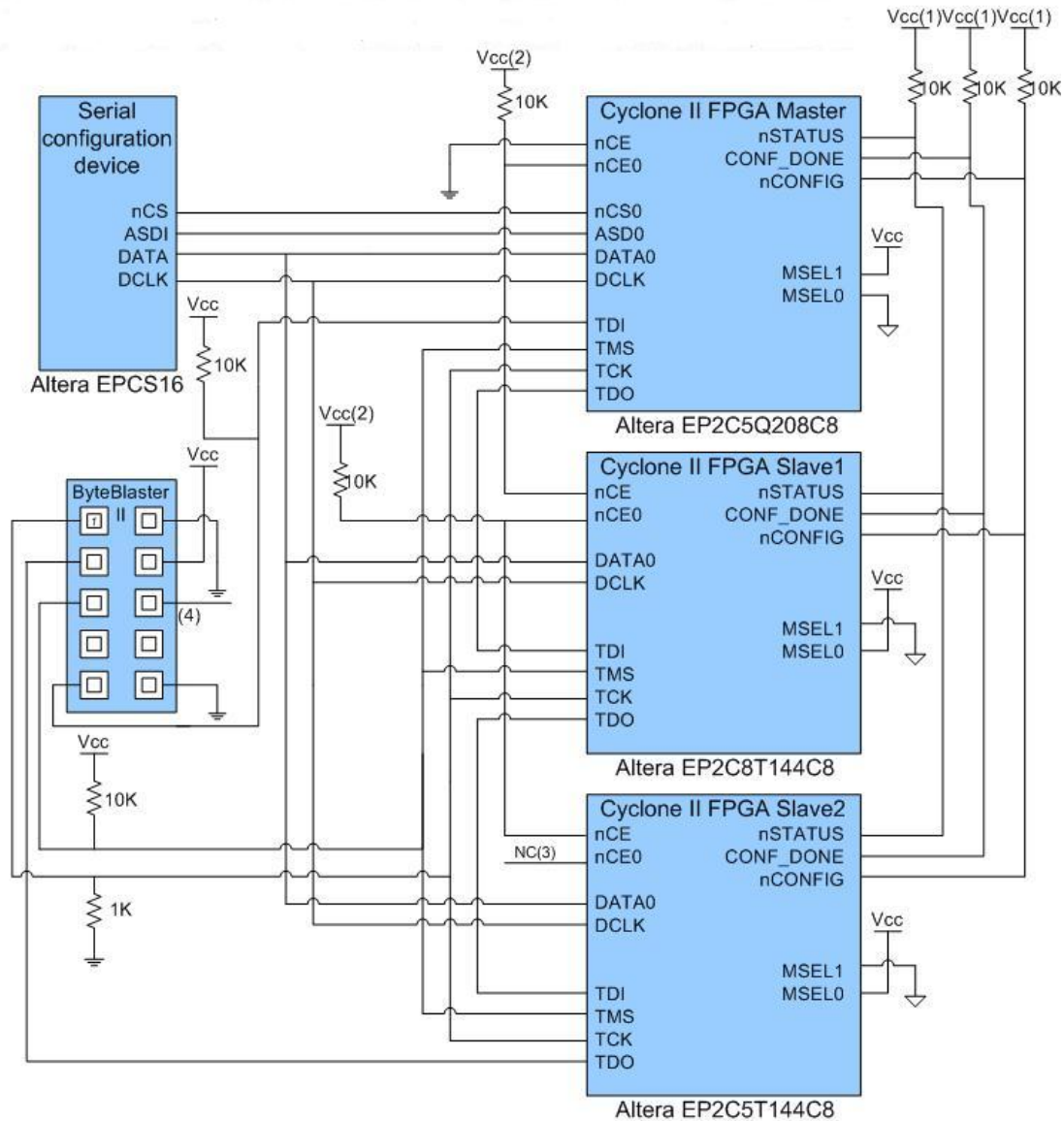


Figure 4.4.: Configuration setup for multiple devices using the JTAG connector and a serial configuration device.

figure 4.4) has been drawn by combining figure 13 – 4 and 13 – 24 in reference [14].

There were some uncertainties while combining these two schematics, as the AS mode has common pull up resistors for each of the status pins (nSTATUS, CONF_DONE and nCONFIG), while the JTAG mode pulled them up separately. The JTAG works however with common pulled up status pins, *if ALL the FPGAs are in user mode. This can be achieved if the serial flash loader contains program*

data for every FPGA or every FPGA in the JTAG programmer chain list are set to be programmed. In AS mode, the entire programming process is interrupted if one fails to retrieve the correct data. The nSTATUS and CONF_DONE pins, which are bi-directional open-drain IOs, will then perform handshakes during programming.

The AS mode requires that the master's MSEL pins are set in AS mode and the slaves in PS mode (see table 4.2 for the MSEL settings). Due to the MSEL settings, the JTAG takes precedence over other MSEL configuration settings, and therefore the settings for the master FPGA can be set as Fast AS mode and the slave FPGAs to PS mode.

Configuration scheme	MSEL1	MSEL0
AS (20 MHz)	0	0
PS	0	1
Fast AS (40 MHz)	1	0
JTAG	1	1

Table 4.2.: The various MSEL settings for the different configuration modes.

The Serial Configuration device must be programmed by help of the Serial Flash Loader. The only disadvantage of using the Serial Flash Loader is that it takes longer to program the device, and there are some file conversions to be done. However, the Serial Configuration device will not be reconfigured that often, so the extra time spent is relatively short in total.

4.4. FPGA master

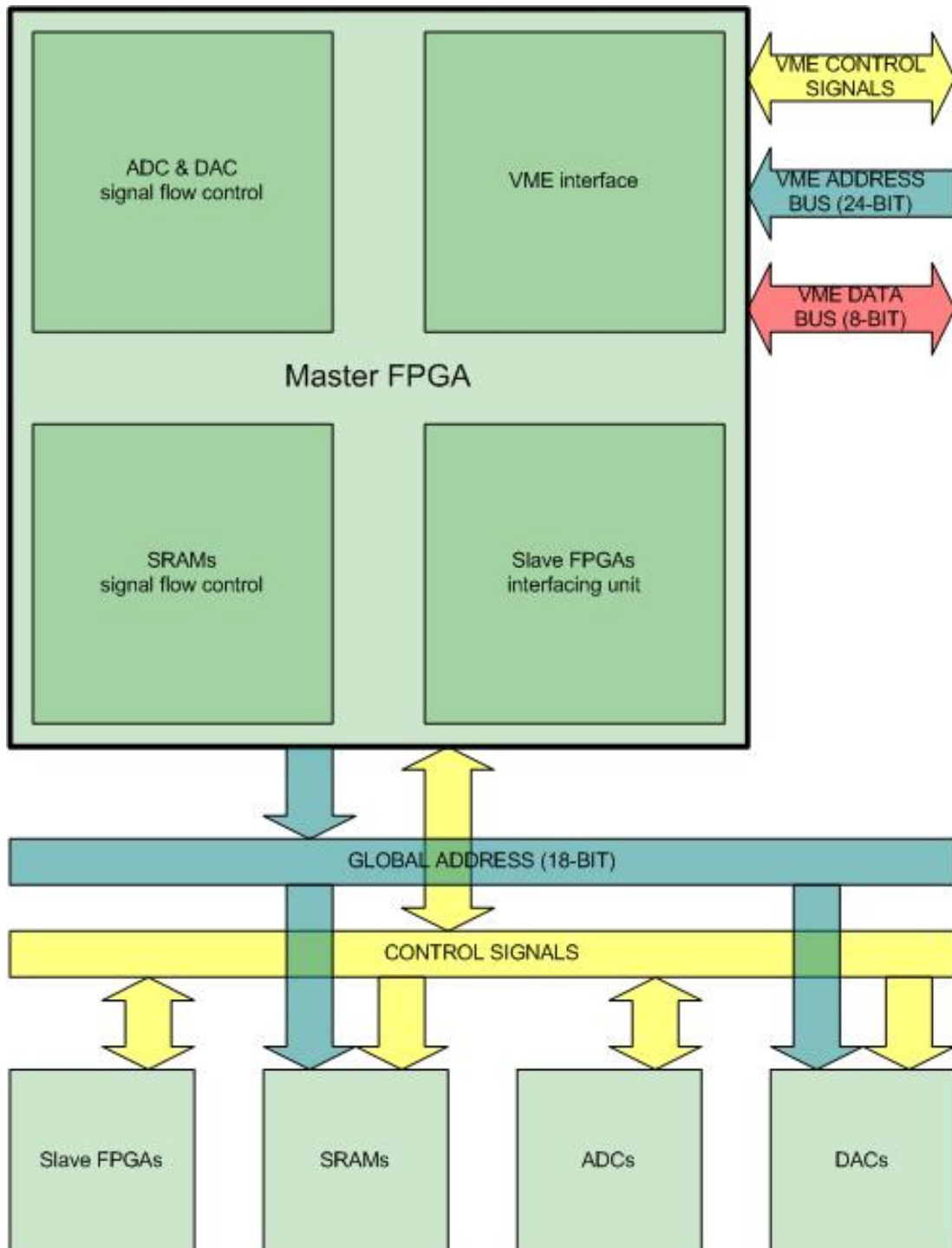


Figure 4.5.: Block overview of the master FPGA system.

4.4.1. VME interface

The VME interface FPGA is the master of the board FPGAs, but in VME-mode it works as a slave for the unit mastering the VME-bus. The VME-bus broadcasts all signals to all connected units, and depending on the address and address modifiers, one responds. If there is no response after a set VME-bus time-out, a bus error will be submitted and the bus released.

VME signal	Description	Bits
A[23..1]	Address bus	24 (23)
AM[5..0]	Address modifier	6
AS _n	Address strobe	1
D[15..0]	Data bus	16 (8)
DS _n [1..0]	Data strobes	2
DTACK _n	Data acknowledge	1
WRITEn	Write	1
SYSRESET _n	System reset	1

Table 4.3.: The listed signals are the connected signals, which are needed for basic VME operation. Signals denoted by n are active low. Though the address bus is defined as a 24-bit bus, it is in reality only a 23-bit bus (as A0 does not exist). The data bus has a width of 16 bit, but is restricted to only 8 bit on the master FPGA due to pin limitations.

There is no A0 address line on the VMEbus. This means that in the software, the VME address has to increment by two for each address, as odd numbers will be approximated down by one. When using the logic analyzer, the read-out address must be multiplied by two, since the A1 address line is considered as the LSB. The A0 address line is however added as ground in the VHDL code to make the software and hardware addresses coincide.

The address modifier defines the addressing mode, which is basically how many address lines should be used, but also which type of access should be made. For the A16 mode, the supervisory access (HEX:2D) and the non-privileged access (HEX:29) have been added. And for the A24 mode, the supervisory access (HEX:3D) and the non-privileged data access (HEX:39). As shown in table 4.4, the A16 mode is used for control mode of the FPGAs and the A24 mode is used for memory read-out. For the A24 mode, an additional BLT¹³(HEX:3B

¹³Block Transfer

Address range (HEX)	Address modifier	Operation
00-1F	A16	FPGA2 chip enable
20-3F	A16	FPGA3 chip enable
40-7F	A16	FPGA1 control
80-8F	A16	DAC control
90-9F	A16	ADC control
00-7FFFF	A24	SRAM read-out

Table 4.4.: Rough VME mapping (see appendix D for the detailed VME memory mapping). A control unit in each of the slave FPGAs will decode their addresses when enabled by the master. Selected memory, of the four, to read out is set in a register.

or 3F) mode has been added. The BLT allows multiple word transfers without re-initiating the VME for each word when reading out from the SRAMs.

A falling edge of the address strobe indicates that the addressing related lines are ready. Hence by this falling edge, the address lines, the address modifier lines and the write signal can be registered. By the falling edge of the data strobes, the data is ready to be sent or received. The actual data transfer to or from the memory map-allocated location will occur when the slave pulls the data acknowledge signal low (the FPGA output pin is active high, as it biases a transistor pulling the data acknowledgement signal low). After the transfer has been fulfilled, the address and data strobes are released (set to inactive high state).

The VME interface has been divided in two entities. The first entity checks the geographical address and address modifier, while the second entity decodes the lower bits according the memory mapping.

The entity that is to verify the addressing, `vme_slotsel_and_dtack`, is shown in figure 4.6. It consists of four dependent processes. At the falling edge of the address strobe, the `REGISTER_ADDR`-process registers the address, geographical address (which is set by a hexadecimal switch on the card), address modifier, write signal and the active low control signal. The control signal is the registered value of `AM4`, as that is the only bit separating the A16 modes from the A24 modes. When a new address modifier has been registered, it is checked by a second process called `AMCHECK`. If the correct values for any of the above mentioned A24 or A16 modes occurs, a verification signal is set high. The third process called `GACHECK` compares the four most significant

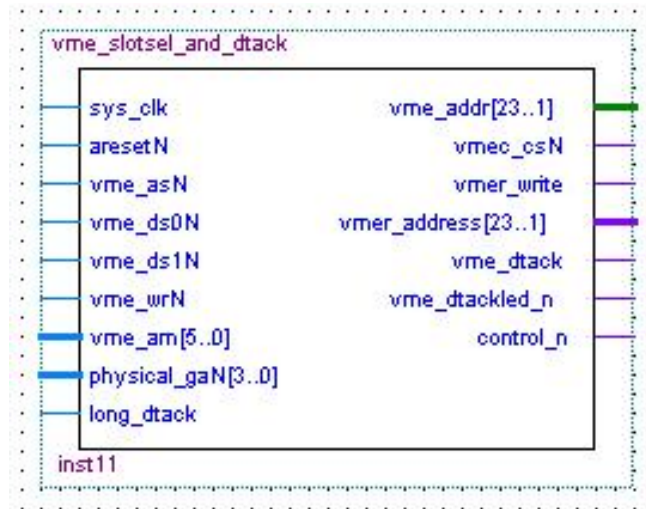


Figure 4.6.: Block symbol for the VME address verifier. Source code can be found in appendix B

address bits $A[23..20]$ with the geographical address set for the card, which should be the same as the VME slot-number to avoid confusion and hence easily address collisions. As we only have a 4-bit physical address, only 16 cards can be uniquely addressed in the VME-crate with this addressing scheme. If one has the invert value of the other (since the binary switch is complementary), the slot address is correct and a signal is set to indicate this. When the correct address modifier and geographical address have been verified, a chip select will be active as long as the address and datastrokes are active (low). The fourth process `DATA_ACKNOWLEDGEMENT` is synchronized by the system clock, and has a state machine which is idle as long as the chip select signal is inactive. When the chip select becomes active, the data acknowledge signal will become active after a delay of four or indefinite (if an ADC is accessed, the delay depends on the conversion time) clock cycles. When the data acknowledgement becomes active, the unit mastering the VMEbus will release the bus and hence also the address and datastrokes. The state machine also has a counter which lights a LED for about a tenth of a second ($2^{22}bits * 25ns = 0,1049s$) initiated by the data acknowledgement.

The second entity, `vme_func_reg`, contains the memory mapping (see table 4.4 for a rough plan of the memory mapping, the detailed memory mapping can be found in appendix D) and selects the appropriate function depending on the address. The entity is enabled by the chip select and control signals from the

vme_slotsel_and_dtack entity and is only active during the period both the chip select and control signals are active.

4.4.2. ADC and DAC signal flow control

The ADC entities made are to interface the read operation of the ADCs to either the SRAM or VME-bus. By looking at the AD7938 timing diagram for reading, one can see that the conversion trigger signal should be held low during the entire conversion. The conversion end is indicated by the falling edge of the busy signal. To recreate the original timing diagram, the idea is to make a one-shot busy signal which sets a conversion start signal that is pulled low by the acquisition clock (see figure 4.7).

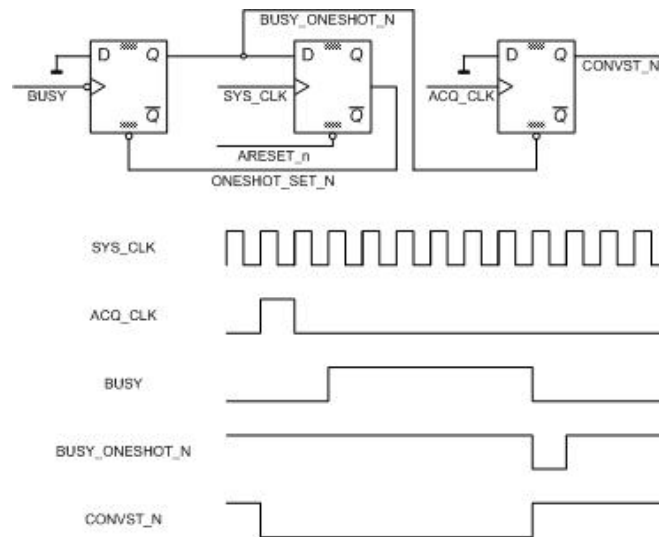


Figure 4.7.: ADC signal flow solution based on the AD7938 timing in read mode.

In the VHDL source code, the read signal flow control has been solved by three processes, one for each flip-flop as they have various clock and set signals. By the conversion-end, a chip select signal delayed by the conversion time is outputted to enable the ADC data output and the SRAM write operation to store the read value.

The AD7484 and AD7677 have the same basic conversion principle as the AD7938, but the conversion trigger should not be held active during the entire conversion. Due to this feature, there is a separate entity to control these two, where the conversion trigger is reset by the rising edge of the busy signal. If

opposite active level for either the busy or conversion signal, this is inverted in the block diagram outside the entity-block.

4.4.3. SRAM signal flow control

When storing the acquisition data, there is a certain delay due to the conversion time of the ADCs. Hence the acquisition clock for the SRAMs storing digitized values have to be delayed until the conversion is over. To solve this, the falling edge of the ADC chip select signal functions as the acquisition trigger for the SRAMs. The ADC chip selects have been set to stay active for four clock cycles, to make sure that the data is available during the SRAM write operation. To make sure the current data is written to the current address, a chip enable delay of one clock cycle has been implemented. A chip enable signal is active for one clock cycle, which is more than the double of the SRAM access time.

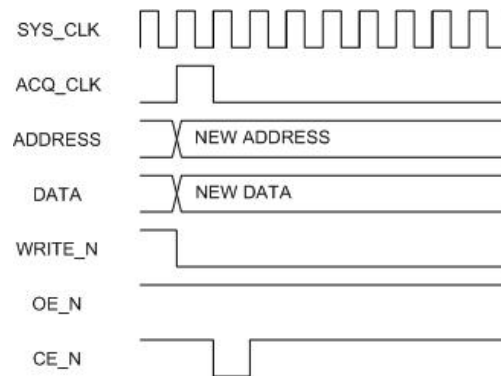


Figure 4.8.: SRAM write signal flow control.

The planned SRAM signal flow for the write mode can be seen in figure 4.8. However, when the acquisition clock is actually the ADC chip select signal, then the address has been incremented earlier by the actual acquisition clock's rising edge.

When reading values from the SRAM, the active SRAM must first be set in the SRAM selection register D1-D0 (see appendix C). The data output enable and write signal therefore depends on whether the read enable is active. The read enable is only active during read-out, which is during VME access in VME24 read mode (see section 4.4.1).

4.5. FPGA slave 1

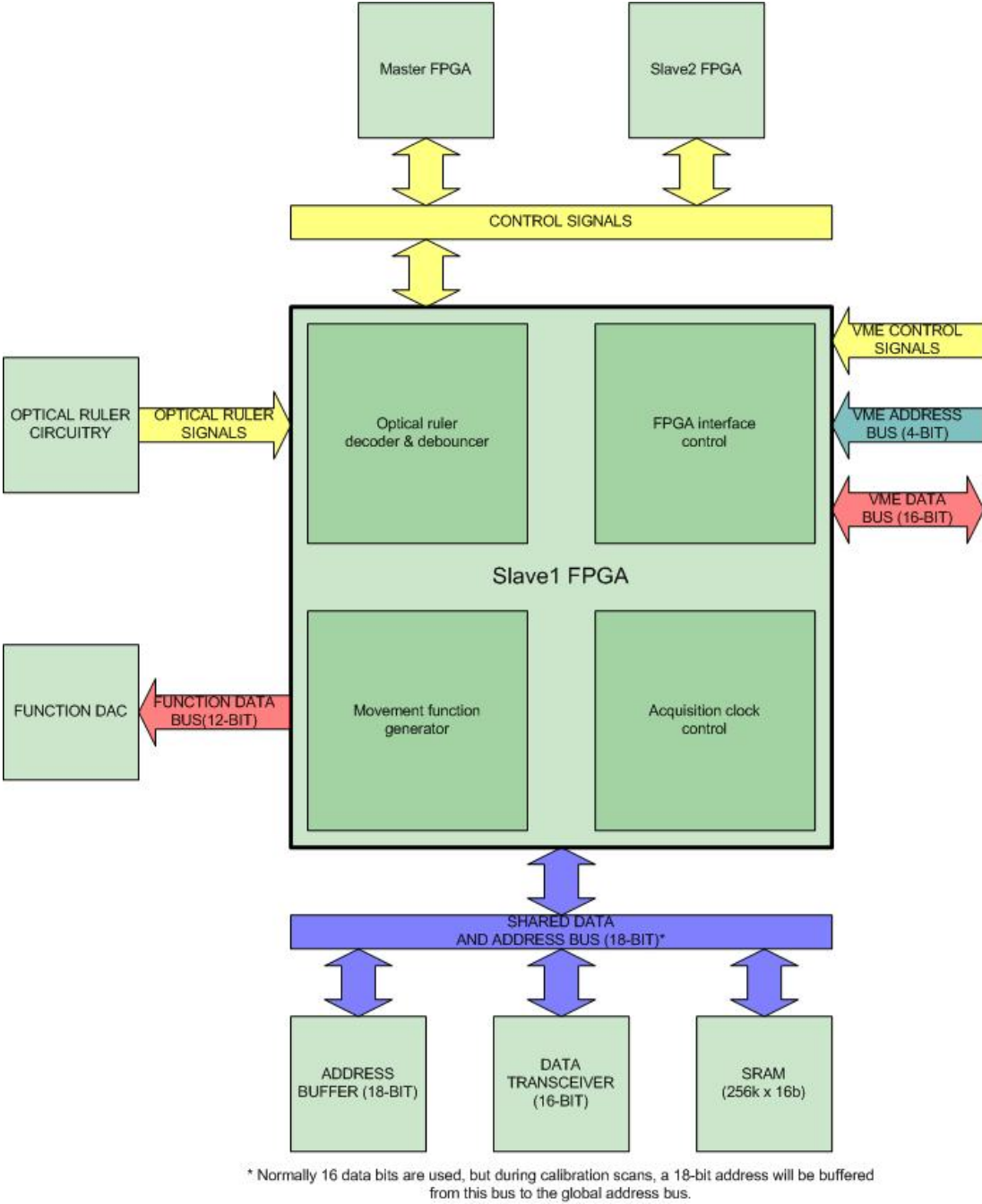


Figure 4.9.: Block overview of the signals connected to the first slave FPGA.

4.5.1. FPGA interface control unit

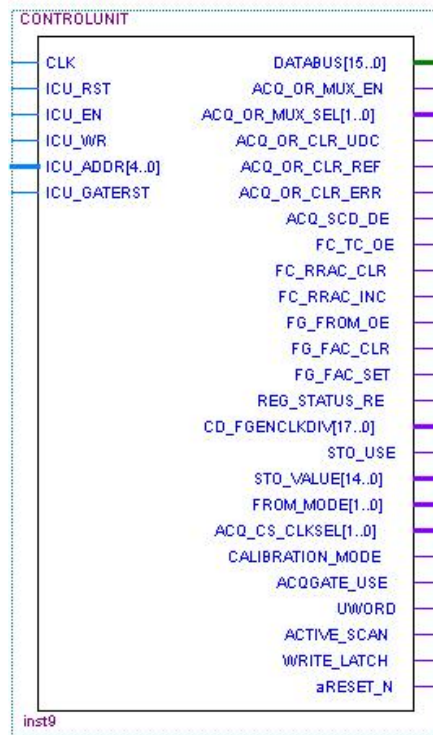


Figure 4.10.: Block symbol for the control unit of the first slave. Most output signals are VME-transferred values stored in the registers.

As the second FPGA is a slave controlled by the master FPGA, a communication interface is essential. The enable signal is set by the master FPGA when control mode VME addresses in the range HEX"00" to HEX"1F" occur. When enabled, the control unit has a memory mapping identical to the VME addressing and it registers the four LSB VME address bits (VME_A[4..1]). The control unit has several registers, from which most set control modes for the wire scanner motion. However there are also registers in other components, which are enabled by a data enable signal output from the control unit when written to or read from the VME. The memory mapping can be seen in table 4.5.

The clock division counter register is an 18-bit register, which sets the division factor of a 40 MHz system clock. The division value will be fed to a clock divider, and the divided clock signal is the clock used to clock the function generator data stored in the function ROMs. This division value is selected depending on the number of values (address width) and the wire scanner velocity. This register is

Control unit memory mapping, write mode		
Address	Data width	Function
00	16bits	Clock division counter register Lo
02	2 bits	Clock division counter register Hi
04	0	Clear quadrature decoder reference register
06	-	-
08	0	Clear quadrature decoder error counter
0A	-	-
0C	0	Clear quadrature decoder U/D counter
0E	16 bits	Acq. Clock division counter register
10	0	Clear Memory address counter
12	0	Set Fgen ROM address counter to FFF
14	0	Clear Fgen ROM address counter
16	12 bits	FGEN end address value
18	8 bits	Control register
1A	0	Start motion
1C	0	Motion reset
1E	0	FPGA II reset (also set by master reset)
Control unit memory mapping, read mode		
Address	Data width	Function
00	16 bits	Clock division counter register Lo
02	2 bits	Clock division counter register Hi
04	16 bits	Ruler Reference Register Lo
06	2 bits	Ruler Reference Register Hi
08	16 bits	Ruler error register Lo
0A	2 bits	Ruler error register Hi
0C	-	-
0E	16 bits	Acq. Clock division counter register
10	-	-
12	16 bits	Status buffer
14	12 bits	FG ROM 4k x 12 bits (incremental per read)
16	12 bits	FGEN end address value
18	8 bits	Control register
1A	-	-
1C	-	-
1E	-	-

Table 4.5.: Memory mapping of the VME-controlled functions of the first slave FPGA enabled by the master FPGA.

stored in the control unit component.

The ruler reference register is the count value of the optical ruler registered by a reference signal when the wire is in mid-position. The ruler error register contains the amount of errors that the EXE610 issues during a scan (see section 4.5.4 for details). Both registers can be read and cleared through the VME-bus. The ruler quadrature decoder up-down counter is a registered counter value which increments/decrements for every μm of the wire movement. This counter should be reset before an in-scan starts. All these registers are stored in the ORQDMUX component which is the optical ruler quadrature decoder and output multiplexer (see section 4.5.4).

The acquisition clock register is a 16-bit register, which sets the division value of the selected acquisition clock (set in control register). This value should be set in accordance with the desired sampling frequency compared to the function generator clock frequency. This register is stored in the acquisition clock divider component, and the control unit enables data during write or read operations through the VME-bus.

The read-memory address is an incremental counter, which increments for every read out from the active ROM. This is to be cleared before the read-outs start, and the amount of read-outs on the VME address for the FG ROM should comply with the address width of the ROM. This is used to check the ROM contents (see section 4.5.3 for details).

The set- and clear-signals for the Fgen address counter are used for transitions between the offsetting function and the fast scan function. For details, see section 4.5.2.

The control register is a 8-bit register, which contains the various mode selection values (see table 4.5.1) and is stored in the control unit. This register can be written and read through the VME-bus.

DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
FGAddMode	Home	AcqGate	ScanMode	CLKS1	CLKS0	FuncMode1	FuncMode0

Table 4.6.: Control register for selecting the various scan modes. Details on the various bits can be found in appendix C.

The start motion is the trigger to start the function generation and the various acquisitions. This trigger sets an active scan signal, which is reset by the gate_rst signal when the scan is over. When the active scan signal is logic high, the

divided clocks for the function generation and acquisitions are output enabled. If the motion is reset, the wire is brought to its initial position through a capacitor discharging circuit.

4.5.2. Function generator

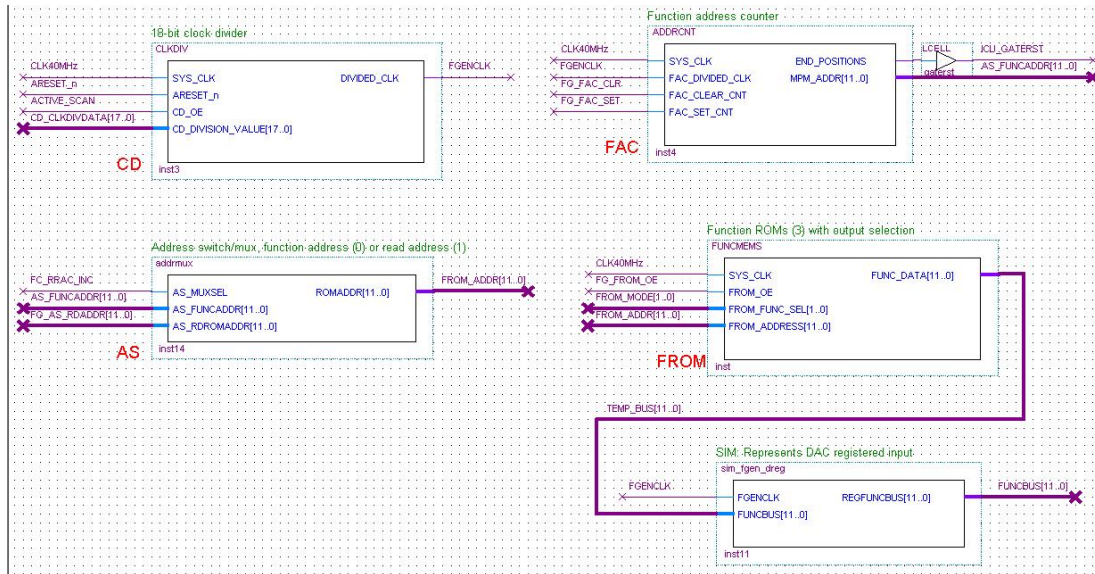


Figure 4.11.: Block schematic of the VHDL top entity of the function generation.

When enabled by the active scan signal from the main control unit in the FPGA, the clock divider counts down from the division value-1 to zero and gives out a clock cycle before loading the division value again. The division register is stored in the main control unit of the FPGA and has 18 bits, which allows a minimum frequency of $\frac{40MHz}{2^{18}} \approx 153Hz$, when dividing the system clock of 40 MHz. When the register holds the clock division value of zero or one, it is disabled, hence the maximum output frequency is half of the input frequency.

The 12-bit output address from the address counter controls the output data by pointing to the ROM readout location. There is a state machine which has two states, each counting in opposite direction. As the state changes, a feedback indicates that the scan is over and outputs a reset to a SR¹⁴-flipflop by its rising edge. This way the active scan signal becomes inactive, and a new scan trigger must occur to activate the movement in opposite direction. Also there are clear

¹⁴Set-Reset

and set signals which can be controlled through the VMEbus. This in order to clear or set the address by the transition between the offset and fast scan profiles.

The memory entity contains three ROMs, where the three different functions (see figure 3.3) are stored. Each memory has $2^{12} = 4096$ words of 12 bits. This means that the FPGA needs at least $2^{12} \cdot 12 \cdot 3 = 147456$ memory bits, which is exactly what the EP2C8T144C8 chip has in it's 36 M4K memory modules (the total RAM bits shown in table 4.1 includes also parity bits). For selecting which profile function should be used, a register value transferred through the VME is stored in the control unit and used as a selection switch input (see table 4.7).

FuncMode[1..0]	Profile
00	Linear offset profile (figure 3.3a).
01	Accelerated and deaccelerated fast scan profile (figure 3.3b).
10	Linear slow scan profile (figure 3.3c).
11	No operation.

Table 4.7.: The register value transferred and stored in the control unit will be set as an input for the profile memory entity. The function is selected by these combinations at the FuncMode[1..0] inputs.

ROM initialization and file generation

The memory entity has three port mapped mega-function ROM memories, each configured to have an address- and data-width of 12 bits. The mega-function has been used in order to allocate data to the actual memory area of the FPGA. The memories will be initialized by MIF¹⁵-files, which have been generated by a matlab-script. Every function is calculated by the means of the address- and data-width, meaning that it is a general solution, and the functions can be generated with different resolutions.

The linear functions are simply generated by the equation $Y(X) = \frac{Y_{PEAK}}{X_{MAX}} \cdot vtX$. Where vtX is a vector with every integer value in the address-width range, Y_{PEAK} is the top amplitude for the given function and X_{MAX} is the top value of the address-width.

For the fast scan profile however, there are three functions to combine. The first part is a square function for the acceleration, then a middle part with a

¹⁵Memory Initialization File

linear function for constant speed while crossing the beam and the end part is an inverted square function for the deceleration. The linear part has been defined to make 25% of the stroke. In addition, an offset of 5% has been added to prevent the fork arm to hit the hard end, as the actual movement may have some overshoot due to the high speed.

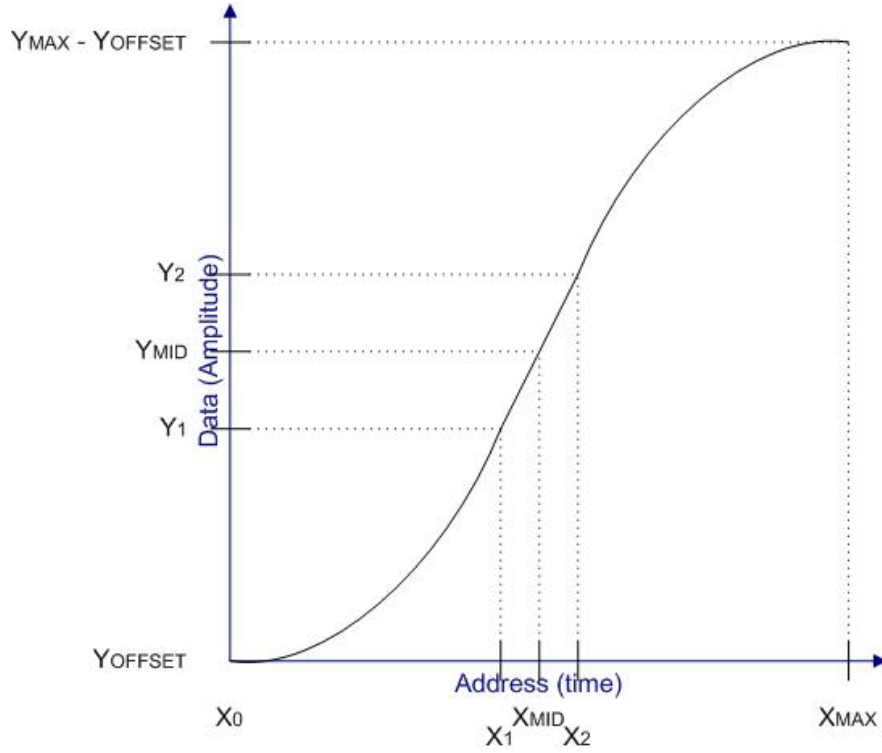


Figure 4.12.: Fast scan set value function.

The three part functions are in principle:

$$Y_{ACCELERATION}(X) = a \cdot X^2 \quad (4.2)$$

$$Y_{CONSTANTSPEED}(X) = b \cdot X + c \quad (4.3)$$

$$Y_{DEACCELERATION}(X) = -a \cdot (X - X_{MAX})^2 + Y_{MAX} \quad (4.4)$$

The variables indicated on figure 4.12 are $Y_{MAX} = 2^{DATAWIDTH}$, $Y_{MID} = \frac{Y_{MAX}}{2}$, $Y_1 = Y_{MID} - \frac{0.25 \cdot Y_{MAX}}{2}$, $Y_2 = Y_{MAX} - Y_1$, $X_0 = 0$, $X_{MID} = \frac{X_{MAX}}{2}$,

$X_1 = \frac{2 \cdot X_{MID} \cdot Y_1}{Y_{MID} + Y_1}$, $X_2 = X_{MAX} - X_1$ and $X_{MAX} = 2^{ADDRESSWIDTH}$. Where X_1 has been found by the two following equations:

$$\frac{\partial Y}{\partial X} = \frac{Y_{MID} - Y_1}{X_{MID} - X_1} \quad (4.5)$$

$$\frac{\partial Y}{\partial X} = 2 \cdot a \cdot X_1 \quad (4.6)$$

By integrating equation 4.6 above, we can find the first parameter a to be $a = \frac{Y_1}{X_1^2}$. The second parameter can be found by specifying that the constant speed equals the speed in the point (X_1, Y_1) , which makes $b = 2 \cdot a \cdot X_1$. This constant speed function has a constant in order to insure a smooth transition to the acceleration part, we already know the Y-coordinate and the constant speed, so the offset becomes $c = Y_1 - b \cdot X_1$.

As we now have all the variables and functions needed, it can be filled in the matlab-script (see appendix E) for generating the three final functions. These can easily be plotted directly, and the vectors printed to files by using program loops.

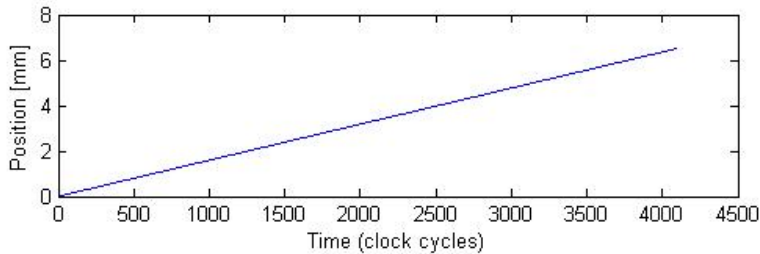


Figure 4.13.: Linear offset set value function as derived in matlab. Y-scale adjusted in accordance with a linear wire scanner (130mm stroke).

4.5.3. Function check

Serious faults can occur if the motion control contains glitches or wrong functions. Therefore the function check has been implemented, to make absolutely sure that the correct functions are stored in the ROMs. The idea is to read out the function ROMs through the VME and display and/or store the data on a computer.

To read out the data, a single address will be used and the address increments for each read-out. There are two entities, one to buffer the function bus data

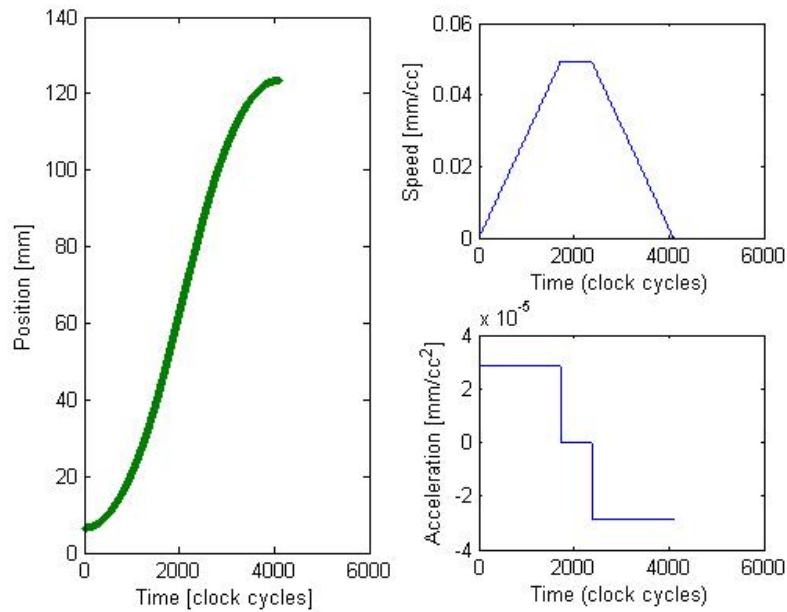


Figure 4.14.: Fast scan set value function, speed and acceleration plotted in matlab. Y-scale adjusted in accordance with a linear wire scanner (130mm stroke).

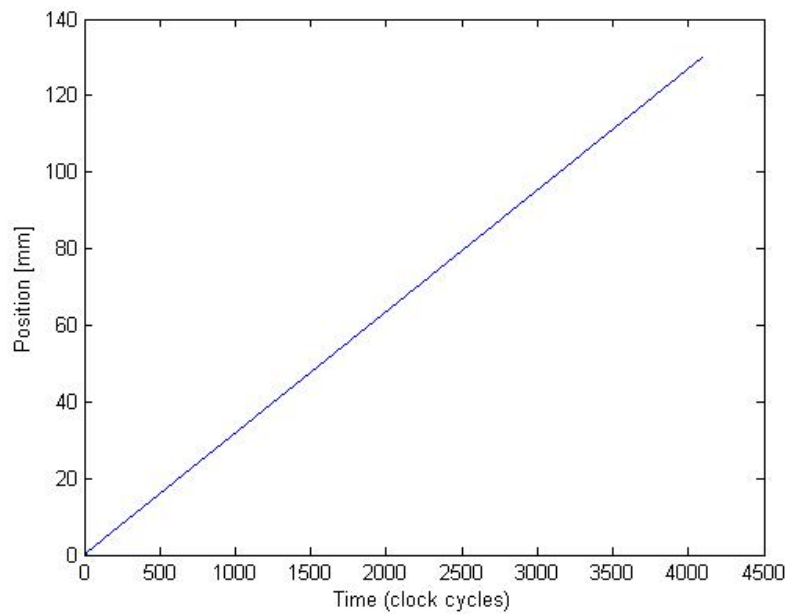


Figure 4.15.: Linear slow scan set value function plotted in matlab. Y-scale adjusted in accordance with a linear wire scanner (130mm stroke).


```

WIDTH=12;
DEPTH=4096;
ADDRESS_RADIX=UNS;
DATA_RADIX=UNS;

CONTENT BEGIN
0 : 205;
1 : 205;
...
...
2046 : 2045;
2047 : 2047;
2048 : 2048;
2049 : 2050;
...
...
4094 : 3890;
4095 : 3890;
END;

```

Figure 4.16.: An extract from the generated .mif-file for the fast scan profile, which has been printed while looping through the matlab generated vectors. The .mif-file is a standard used by Altera, where data-width and address-depth must be defined. The contents are listed with the address first, followed by the corresponding data.

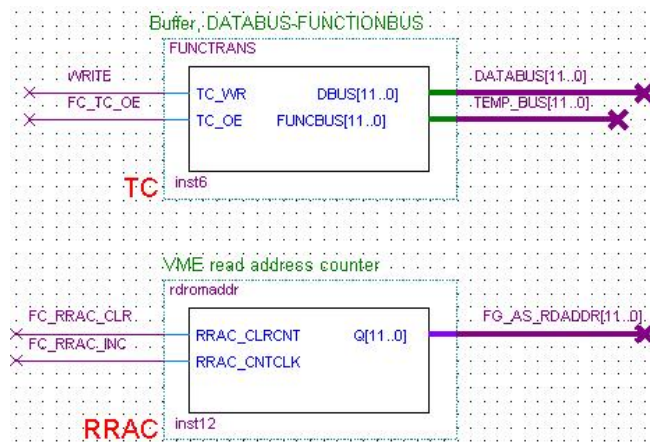


Figure 4.17.: Block schematic of the function check entities.

and one to increment the address. Before the first read-out is done, a clear signal should be sent through the VME. To perform the clear, a write signal is sent to VME address HEX:14. When reading from the ROM, the number of read-outs should be set to the same depth as the memory addresses. The read-out is performed by a read signal on the same VME address as the clear signal, HEX:14. During the active read-out, an output enable signal is set for both the

buffer entity and the address counter. The buffer will output the function data to the VME data bus, and the address counter will increment at the falling edge of the output enable (otherwise address 0 is not accessible). Also the output enable signal switches the address switch (see figure 4.11) for the function ROM, to access the address given by the read-out counter.

4.5.4. Calibration

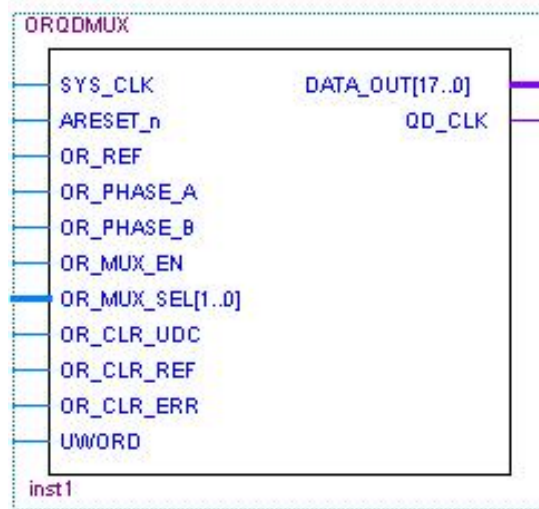


Figure 4.18.: Block symbol for the optical ruler quad decoder

As mentioned in section 3.2.1, for the LHC it is necessary to calibrate the potentiometer with an optical ruler to improve the linearity. By creating a look-up-table using the optical ruler as the address source and the potentiometer as the data source, it can later be checked in software which actual value the potentiometer has with an accuracy of $1\mu m$.

As explained in 3.2.1, the 10-fold interpolation is needed to achieve an accuracy of $1\mu m$, where one period indicates a movement of $4\mu m$ as there are four different states between the two phases, each indicating a movement of $1\mu m$. The four different states are defined in table 4.8.

One state alone does not indicate whether the direction is forward or reverse. Both the previous state and the current state are needed to detect if the counter should increment or decrement. The simplest way to solve this is by using a state machine. The state machine has been defined by 5 states, one initialization

State	Phase A	Phase B
1	0	0
2	0	1
3	1	0
4	1	1

Table 4.8.: The four different phase states of the optical ruler.

state and the four phase states found in table 4.8. The state machine block schematic can be seen in figure 4.19 and the conditions are shown in figure 4.20. In the initialization state, only the current state is detected to find out the start position. No increments or decrements occur during this initialization step. When the phase states are active however, the phase conditions will be checked by every rising edge of the system clock. If the phase state changes, a counter will either increment or decrement depending on the phase change. A forward motion will result in the incremental route (counter clock wise in figure 4.19), while a reverse motion results in the decrementing route (clock wise in figure 4.19). However, if both phases toggle, an error counter will increment and the initialization state checks the current active phase state.

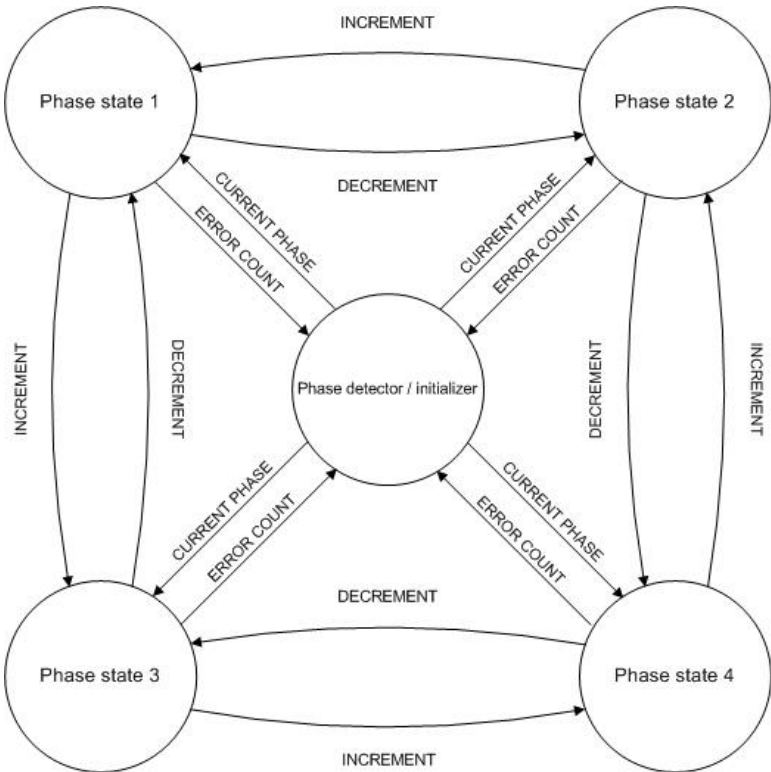


Figure 4.19.: Optical ruler quad decoder state machine.

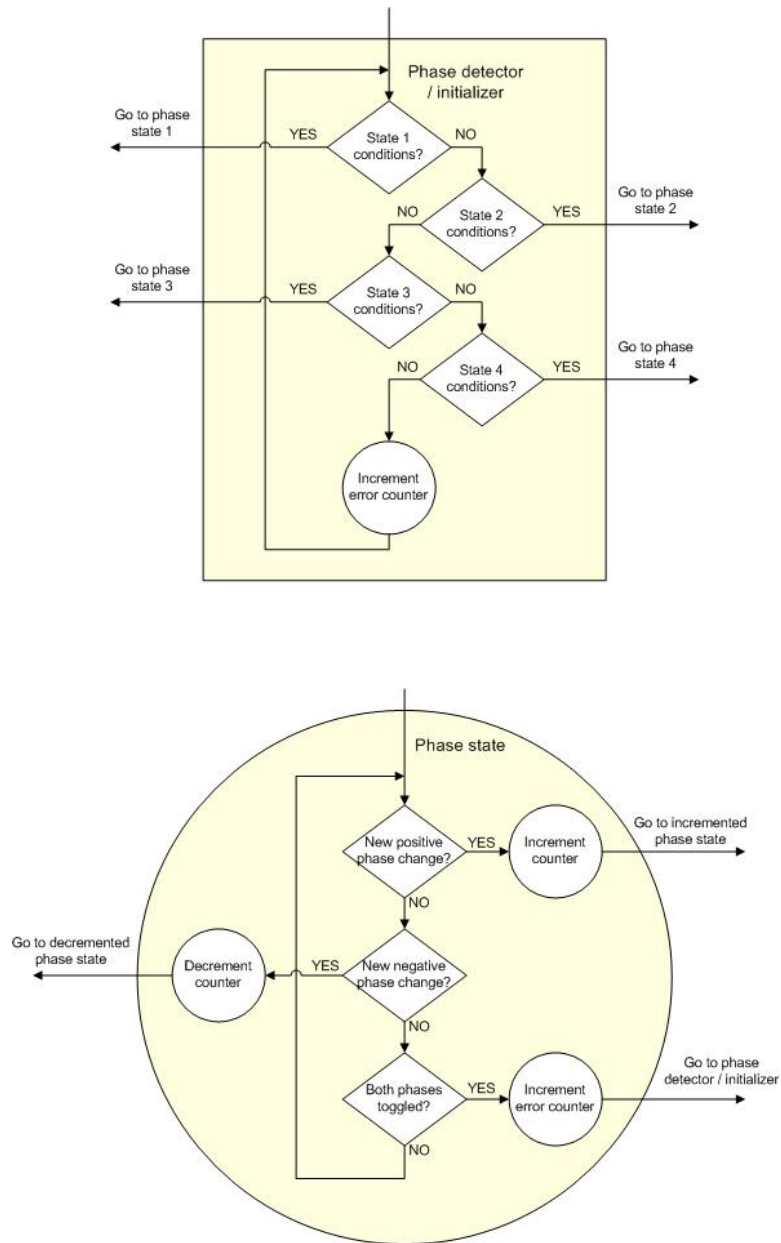


Figure 4.20.: Conditions to change state in the optical ruler quad decoder state machine.

4.6. FPGA slave 2

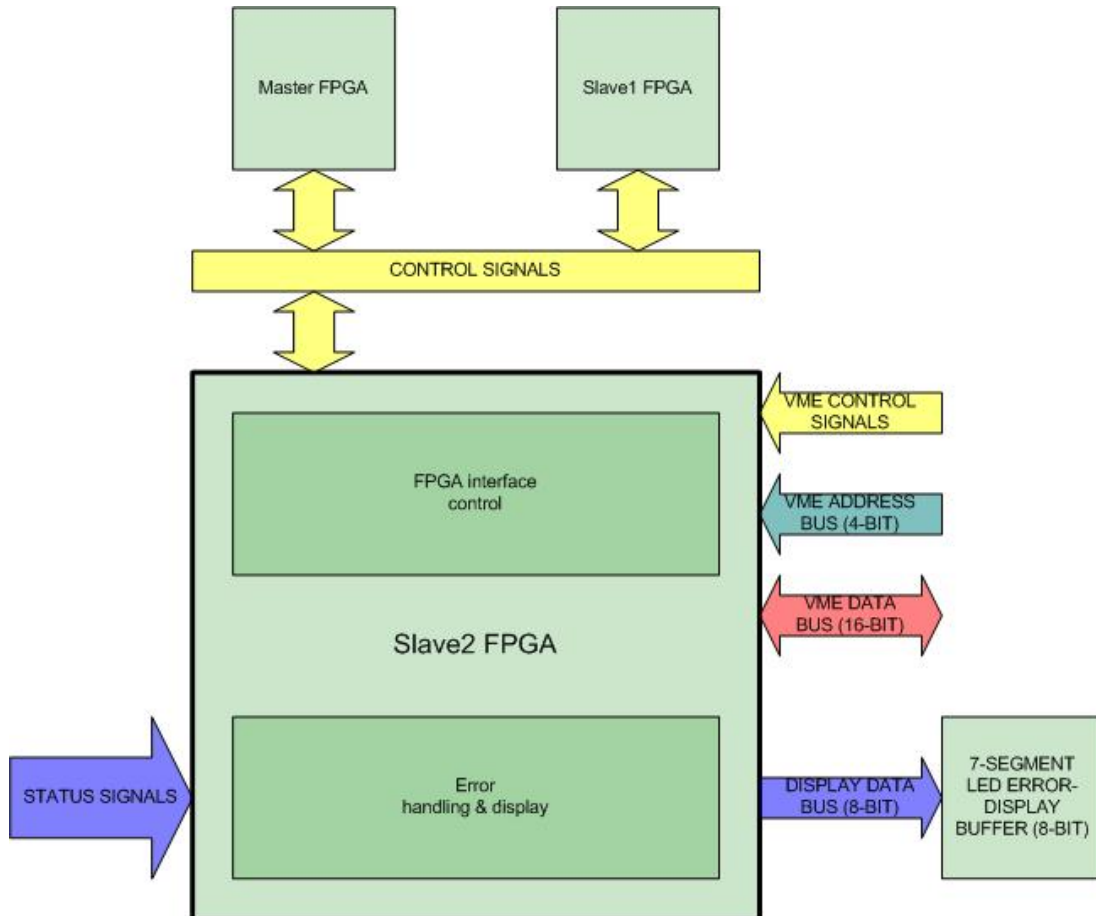


Figure 4.21.: Block overview of the signals connected to the second slave FPGA.

4.6.1. FPGA interface control unit

The control unit works in the same way as in the first slave FPGA, and is enabled by the master FPGA when the assigned VME-address range from HEX:20 to HEX:3F occurs. As seen in Figure 4.22, this is however a much smaller unit with much fewer outputs. The memory mapping currently only consists of three addresses. There is one to read the low word of the error register (HEX:20), one to read the high word of the error register (HEX:22) and one to write a reset signal (HEX:3F) to reset the error register and hence error messages. The error lines that are read out are fed by a status-register, which samples comparator

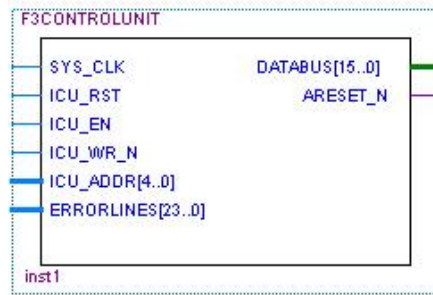


Figure 4.22.: Block symbol for the control unit in slave two.

values giving logic high if no errors and logic low if errors occur. The reset signal also releases a scan-inhibit signal which prevents a scan to start when active due to detected errors.

4.6.2. Error display

This is a register used to indicate if errors have occurred on the card. A bunch of comparators have been used to indicate whether signals have unusual voltage levels. If the comparator gives a logic low, then this indicates an error and this must be registered and displayed. To hold the error bits where an error has occurred, a feedback of the register output feeds an AND-gate on the input signal (see figure 4.23). Thus an logic low will be held until all bits are set high by the asynchronous reset signal.

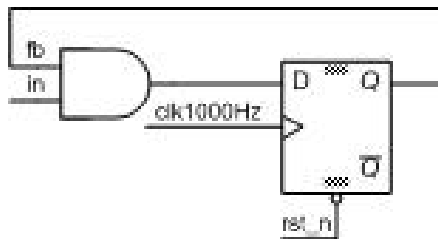


Figure 4.23.: Holding registered errors until the asynchronous reset is activated.

The error sampling of the status lines has been set to have a sampling rate of 1000 Hz. When any error occurs, a scan inhibit signal must be set to prevent a new scan to start while there are errors. With a 1000 Hz sampling frequency, the maximum time spent from an error occurs until it is registered is close to 1 ms. This should be sufficient to avoid new scans to be started before the scan inhibit

is set in case of errors.

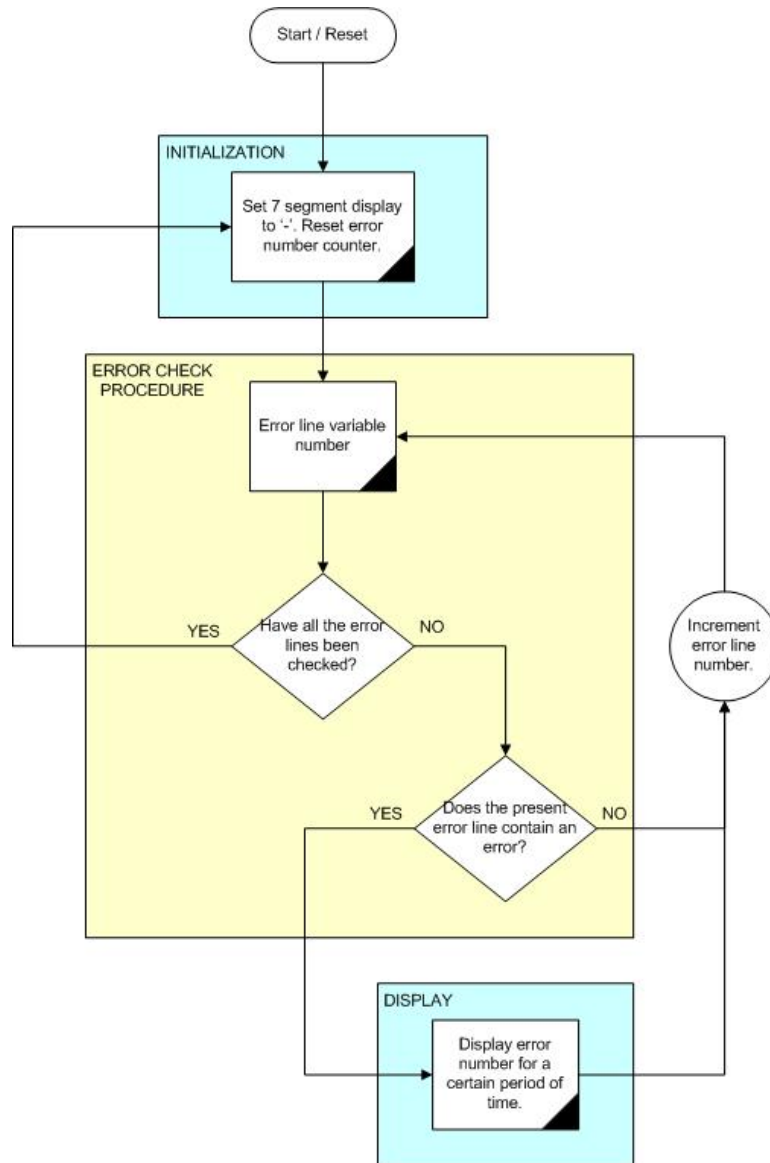


Figure 4.24.: State machine controlling the seven segment LED display.

The occurred errors are also displayed on a seven segment display. To display the errors, a state machine (see figure 4.24) checks every error bit and displays those that are logic low as an error character on the seven segment LED display. The hexadecimal codes for the displayed error characters have been stored in a data array, where the array cells coincide with the error bit-numbers.

Error messages	
<i>Error description</i>	<i>Display</i>
No error.	-
Wire broken.	0
Wire hot.	1
Heat sink over temperature.	2
Fan current low.	3
-2V VME supply missing.	4
-2V internal supply missing.	5
-5V internal supply missing.	6
MPS voltage error.	7
Wire1 not near home position.	8
Wire2 not near home position.	9
Wire3 not near home position.	A
Wire4 not near home position.	b
Potentiometer1 voltage too low.	C
Potentiometer2 voltage too low.	d
Potentiometer3 voltage too low.	E
Potentiometer4 voltage too low.	F
Ruler status bad.	I
Internal FPGA1 voltage (PG1) bad.	J
PLL1 voltage FPGA1 (PG11) bad.	L
PLL2 voltage FPGA1 (PG12) bad.	M
Internal FPGA2 voltage voltage PG2 bad.	n
PLL1 voltage FPGA2 (PG21) bad.	o
PLL2 voltage FPGA2 (PG22) bad.	P
PLL1 voltage FPGA3 (PG31) bad.	r
PLL2 voltage FPGA3 (PG32) bad.	t

Table 4.9.: A list of error messages.

7 segment LED decoder, common anode									
Character	Segment								Code (HEX)
	a	b	c	d	e	f	g	dp	
A	0	0	0	1	0	0	0	1	11
b	1	1	0	0	0	0	0	1	C1
C	0	1	1	0	0	0	1	1	63
d	1	0	0	0	0	1	0	1	85
E	0	1	1	0	0	0	0	1	61
F	0	1	1	1	0	0	0	1	71
g (9)	0	0	0	0	1	0	0	1	09
H (K,X)	1	0	0	1	0	0	0	1	91
I	1	1	1	1	0	0	1	1	F3
J	1	0	0	0	0	1	1	1	87
K (H,X)	1	0	0	1	0	0	0	1	91
L	1	1	1	0	0	0	1	1	E3
M	0	1	0	1	0	1	1	1	57
n	1	1	0	1	0	1	0	1	D5
o	1	1	0	0	0	1	0	1	C5
P	0	0	1	1	0	0	0	1	31
Q	0	0	0	1	1	0	0	1	19
r	1	1	1	1	0	1	0	1	F5
S (5)	0	1	0	0	1	0	0	1	49
t	1	1	1	0	0	0	0	1	E1
U	1	0	0	0	0	0	1	1	83
v	1	1	0	0	0	1	1	1	C7
W	1	0	1	0	1	0	1	1	AB
X (H,K)	1	0	0	1	0	0	0	1	91
Y	1	0	0	0	1	0	0	1	89
Z (2)	0	0	1	0	0	1	0	1	25
0	0	0	0	0	0	0	1	1	03
1	1	0	0	1	1	1	1	1	9F
2 (Z)	0	0	1	0	0	1	0	1	25
3	0	0	0	0	1	1	0	1	0D
4	1	0	0	1	1	0	0	1	99
5 (S)	0	1	0	0	1	0	0	1	49
6	0	1	0	0	0	0	0	1	41
7	0	0	0	1	1	1	1	1	1F
8	0	0	0	0	0	0	0	1	01
9 (g)	0	0	0	0	1	0	0	1	09

Table 4.10.: The common anode seven segment LED display driver codes in hexadecimal numbers.

5. Measurements on the prototypes

5.1. FPGA prototype

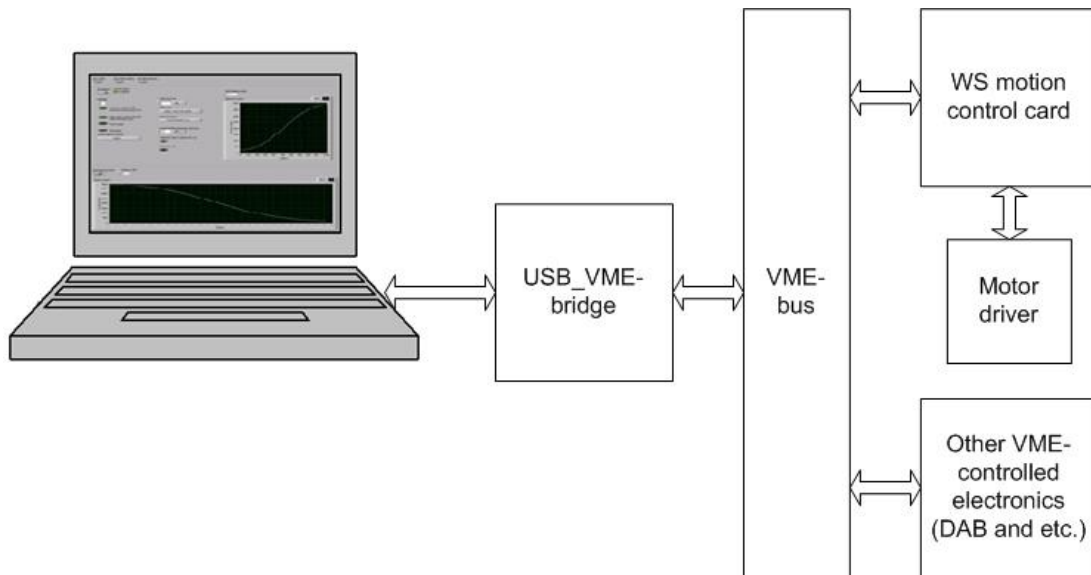


Figure 5.1.: Measurement setup for the FPGA prototype. As the wire scanner is currently dissembled and some of the collaborating cards are not available, simulation values were fed to the memories. The FPGA prototype aims to check the programming hardware, memory storage and VME transfers.

While waiting for the complete analogue design, it was decided that a FPGA prototype card should be made to advance with the digital design. Only one JTAG connector is foreseen on the front-panel due to space limitations, and therefore the serial flash loader would be an useful tool to program the serial configuration device through the master FPGA. There will however be another download connector directly on the ASMI for the serial configuration device on this FPGA prototype card, this to be able to compare the configuration-reliability and -time with the serial flash loader setup through the JTAG connector.

The main purpose of this card is to test the above mentioned configuration scheme and some main features of the system functionality. Most of the analogue circuits were excluded in this prototype, which is why it has been named the FPGA prototype. The complete schematics and the PCB-routing can be found in the appendix H.

5.1.1. Configuration setup

A configuration scheme is needed to program the FPGAs. As the program is stored in a volatile memory in a FPGA, an additional non-volatile flash memory which fills the FPGAs is needed.

It is desired to test the configuration and debug tools for several FPGAs without multiple connectors. The idea is to use the JTAG interface to program and debug the FPGAs directly, and to download the program to the serial configuration device through the master FPGA using a serial flash loader.

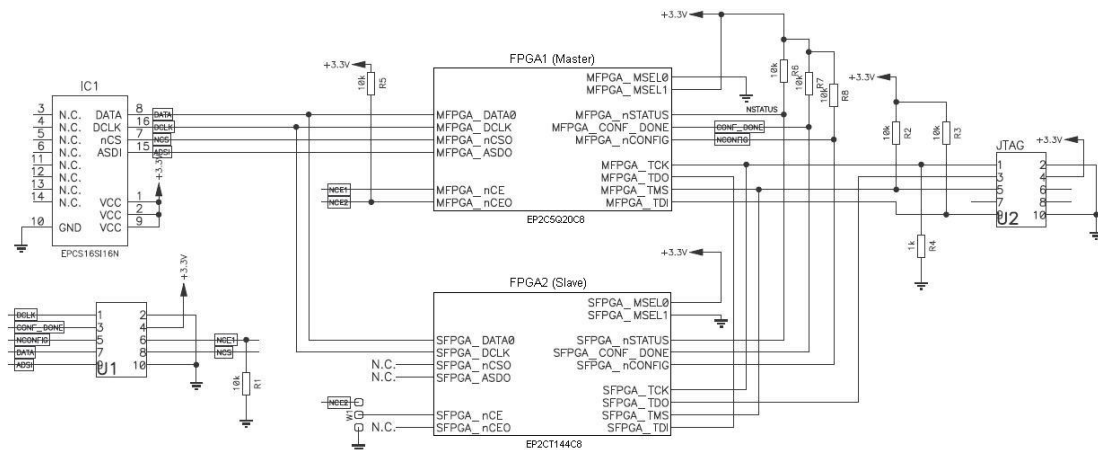


Figure 5.2.: Configuration scheme for the two FPGAs.

The configuration schematic (see figure 5.2) is a result of three assembled configuration setups (see reference [14] figure 13-4 and 13-24, and reference [15] figure 9-5). As every FPGA in a multiple FPGA JTAG chain has nCE pulled to ground and the status pins CONF_DONE and nSTATUS are pulled up separately for each FPGA, some jumpers have been added to set the slave nCE of the slave FPGA to ground and change between common pull-ups and separate pull-ups.

Measurements

The JTAG programming mode works for single FPGAs either when the status pins are pulled up separately or when all FPGAs are in user mode (as nSTATUS and CONF_DONE have a high state in user mode). Before the FPGAs have been programmed, the CONF_DONE has a low state and tries to get configured through the ASMI-interface with the serial configuration device.

For the FPGAs to enter user mode, they must all be configured by the serial configuration device at power up or by pulling down nCONFIG, or the JTAG mode must be set to program every FPGA in the chain list. So to fill the serial configuration device when only a JTAG connector is available, the first FPGA configuration must include a .sof¹-file for each FPGA in the chain list. After configuration, the master FPGA can be programmed individually, hence the flash loader can be downloaded to fill the serial flash memory. The .jic²-file loaded to the flash memory in the serial configuration device should then include program data for all the FPGAs in the chain. After the flash memory is filled, every FPGA will be filled at power-up or by pulling nCONFIG low, and then enters user mode. Thus the JTAG mode works for each FPGA even when the status pins are commonly pulled up and the nCE is not grounded.

This concludes that the jumpers can be left out and the nCE and status pins are set according the settings for an active serial configuration (otherwise the serial configuration device does not work).

5.1.2. VME circuitry

The VME-bus is used to interface between cards within the same VME crate and communicate with remote systems. The VME-bus is one of the standard protocols used at CERN for communicating with the electronics equipment.

The VME rack uses the extended 64 bit VMEbus called VME64x. In this project however, only 16 bit data transfers and 24(23) bit addresses will occur. Hence the address- and data signals on the J2/P2 connector will not be used, and as the VME interface is simplified as well, the VME part includes a total amount of 71 signals (see table 5.1) on the FPGA prototype card. For setting the card's

¹SRAM object file

²JTAG indirect configuration

Signal	Connector	Signal	Connector	Signal	Connector
A01	J1-A30	AM1	J1-B17	DTACK*	J1-A16
A02	J1-A29	AM2	J1-B18	SYSRESET*	J1-C12
A03	J1-A28	AM3	J1-B19	WRITE*	J1-A14
A04	J1-A27	AM4	J1-A23	BUNCHSEL0	J0-D12
A05	J1-A26	AM5	J1-C14	BUNCHSEL1	J0-D13
A06	J1-A25	AS*	J1-A18	BUNCHSEL2	J0-D14
A07	J1-A24	D00	J1-A01	BUNCHSEL3	J0-D15
A08	J1-C30	D01	J1-A02	BUNCHSEL4	J0-D16
A09	J1-C29	D02	J1-A03	BUNCHSEL5	J0-D17
A10	J1-C28	D03	J1-A04	BUNCHSEL6	J0-D18
A11	J1-C27	D04	J1-A05	BUNCHSEL7	J0-D19
A12	J1-C26	D05	J1-A06	BUSLINE0	J0-A12
A13	J1-C25	D06	J1-A07	BUSLINE1	J0-A13
A14	J1-C24	D07	J1-A08	BUSLINE2	J0-A14
A15	J1-C23	D08	J1-C01	BUSLINE3	J0-A15
A16	J1-C22	D09	J1-C02	BUSLINE4	J0-A16
A17	J1-C21	D10	J1-C03	BUSLINE5	J0-A17
A18	J1-C20	D11	J1-C04	BUSLINE6	J0-A18
A19	J1-C19	D12	J1-C05	BUSLINE7	J0-A19
A20	J1-C18	D13	J1-C06	CLK+	J0-E16
A21	J1-C17	D14	J1-C07	CLK-	J0-E17
A22	J1-C16	D15	J1-C08	TCD+	J0-E12
A23	J1-C15	DS0*	J1-A13	TCD-	J0-E13
AM0	J1-B16	DS1*	J1-A12		

Table 5.1.: VME-bus and user signals used on the prototype card. * annotates an active low signal.

base address in the rack, a binary coded hexadecimal 16 position rotary switch has been added, leaving four address signals for address comparison.

The VME signals have to be buffered close to where they enter the PCB, in order to minimize capacitance of signal lines. This is why the SN74LVC541APW 8-bit buffers have been added to every VME output signal and the SN74LVC4245ADW 8-bit transceivers have been added to every bidirectional signal. The buffers are always enabled, but the direction and output enable inputs of the transceivers will be controlled by the master FPGA, depending on the addressed function. The data acknowledge signal sinks through a transistor since it requires a current that exceeds the FPGA current limit and it is therefore switched by an output pin from the FPGA.

Measurements

The VME-bus can be accessed either through the USB-port connected to a CAEN VX1718 USB-VME bridge or through RS-232 or Ethernet protocol to the PowerPC. As the software provided with the CAEN VX1718 has a simpler user interface and is faster than the RS-232, the USB-VME bridge has been privileged during these tests.

It has been measured that one single VME transfer cycle with bus request takes about 740 ns. The data acknowledge transistor spends 140 ns, and the set data acknowledge delay from the falling edge of the data strobe is 100 ns. Hence about one third is due to the actual transfer handshake, while two thirds are used to request and release access to the bus.

If the idle time between the VME accesses was as little as say 60 ns, then a maximum achieved transfer data rate would be:

$$2\text{Bytes} \cdot \frac{1}{800 \cdot 10^{-9}\text{s}} = 2.5\text{MBps} \quad (5.1)$$

However, with a read-out loop with no delay set in the software, the bus request has been measured to have a frequency of 2.8 kHz. Thus the actual achieved transfer data rate was:

$$2\text{Bytes} \cdot 2.8\text{kHz} = 5.6\text{kBps} \quad (5.2)$$

Which is approximately 450 times slower than the theoretic transfer rate. The reason is due to re-initialization of the USB interface for each word readout, which demands extra commands and hereby time. To solve this, the block transfer mode has been introduced. When using the block transfer, a memory is reserved on the USB-VME bridge. The USB-VME bridge accesses the VME-bus continuously while incrementing the address and pulling the data strobes low for each transfer, then the data is sent over the USB as a bigger package. By doing this, the readout time of one SRAM (512 kB) was reduced from 91.4 seconds to 0.3 seconds. Thus the transfer data rate had been improved to:

$$\frac{512\text{kBytes}}{0.3\text{s}} = 1.7\text{MBps} \quad (5.3)$$

Which is very close to the theoretical transfer rate found in equation 5.1.

5.1.3. Power supplies

Signal	Connector
+5V	J1-A32
+5V	J1-B32
+5V	J1-C32
+3.3V	J1-D12
+3.3V	J1-D14
+3.3V	J1-D16
+3.3V	J1-D18
+3.3V	J1-D20
+3.3V	J1-D22
+3.3V	J1-D24
+3.3V	J1-D26
+3.3V	J1-D28
+3.3V	J1-D30

Table 5.2.: VME power supply connector pins.

The circuits on the FPGA prototype card only need +3.3V, +5V and 1.2V. The +3.3V and +5V power supplies will be provided through the VME connectors (see table 5.2), while the 1.2V is regulated by a low voltage drop regulator, LMS5258MF-1.2, as shown in Figure 5.3. A regulator will be placed close to every FPGA, as the +1.2V is for the internal voltages of the FPGAs. The input voltage of the low voltage drop regulator will be +3.3V, and a $1\mu\text{F}$ ceramic capacitor will be placed on the output voltage of 1.2V for stability. The regulator also have a Power Good indicator, but this is not needed as the Cyclone II FPGA has an internal POR³ circuit surveying the voltage levels. A $100k\Omega$ pull-up resistor has still been drawn up from the Power Good output in case other circuit needs a POR safety at some later stage. Both +3.3V and +5V are available directly and will be decoupled to ground through some $100\mu\text{F}$ capacitors in order to filter low-frequency noise and supply extra current when many outputs switch simultaneously.

Measurements

The voltages have been measured to be quite accurate and having insignificant ripple. The measured currents were less than 0.2 A, which is no problem for the

³Power-On-Reset

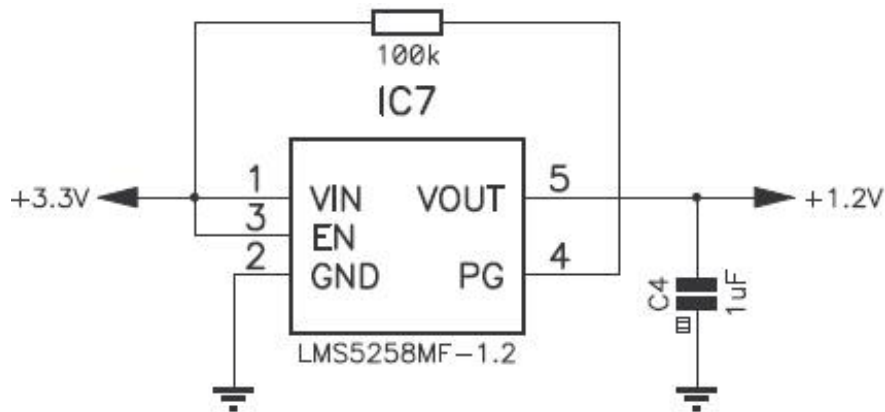


Figure 5.3.: Low voltage drop-out regulator creating a 1.2V power supply from a 3.3V voltage source.

VME crate to feed.

5.1.4. Optical ruler circuitry

The optical ruler is needed to create the calibration-LUT, which is used to improve the potentiometer's accuracy to $1\mu m$.

The optical ruler inputs will be connected to a SubD15 male connector (see figure 5.4). There are 15 pins, whereas 14 of them are used (see table 5.3) on the prototype card.

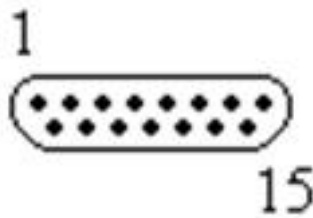


Figure 5.4.: SubD15 male connector.

The ruler phases (here called A and B) represent the two phases which indicates the movement of the wire-scanner's fork arm (see section 3.2.1 for details). The phases are differential signals, but by using the Quad Differential Line Receiver DS26C32ATM, the FPGA inputs are fed with single ended signals.

Pin	Signal
1	Ruler phase A+
2	Ruler phase B+
3	Ruler zero-reference+
4	Ruler status+
5	Vcc +5V
6	Ruler MUX0
7	Ruler MUX1
8	GND
9	Ruler phase A-
10	Ruler phase B-
11	Ruler zero-reference-
12	Ruler status-
13	GND
14	Vcc +5V
15	not connected

Table 5.3.: Pinning specification of the SubD15 male connector for the optical ruler.

Measurements

By the time the FPGA prototype was produced, the SOP⁴ package of the DS26C32ATM component was still not available and the linear wire scanner used for testing had been disassembled due to some modifications. The wire scanner is still not reassembled at this time, but some tests were done at an earlier stage using a Microtronix Stratix Development Kit using an Altera Stratix EP1S25 FPGA and an external dual in-line Quad Differential Line Receiver.

The tests were done by manual scans, which does not achieve the actual speed. The optical ruler counter worked fine and without any error counts during the motion, but error counts occurred when the wire scanner was stalled. This problem can however be solved by resetting the error counter by the scan trigger or by use of an acquisition gate.

5.1.5. SRAM circuitry

The SRAMs are needed to store all the acquired data. As it is important to check that the SRAM interface works properly, one of the SRAM modules has

⁴Small Outline Package

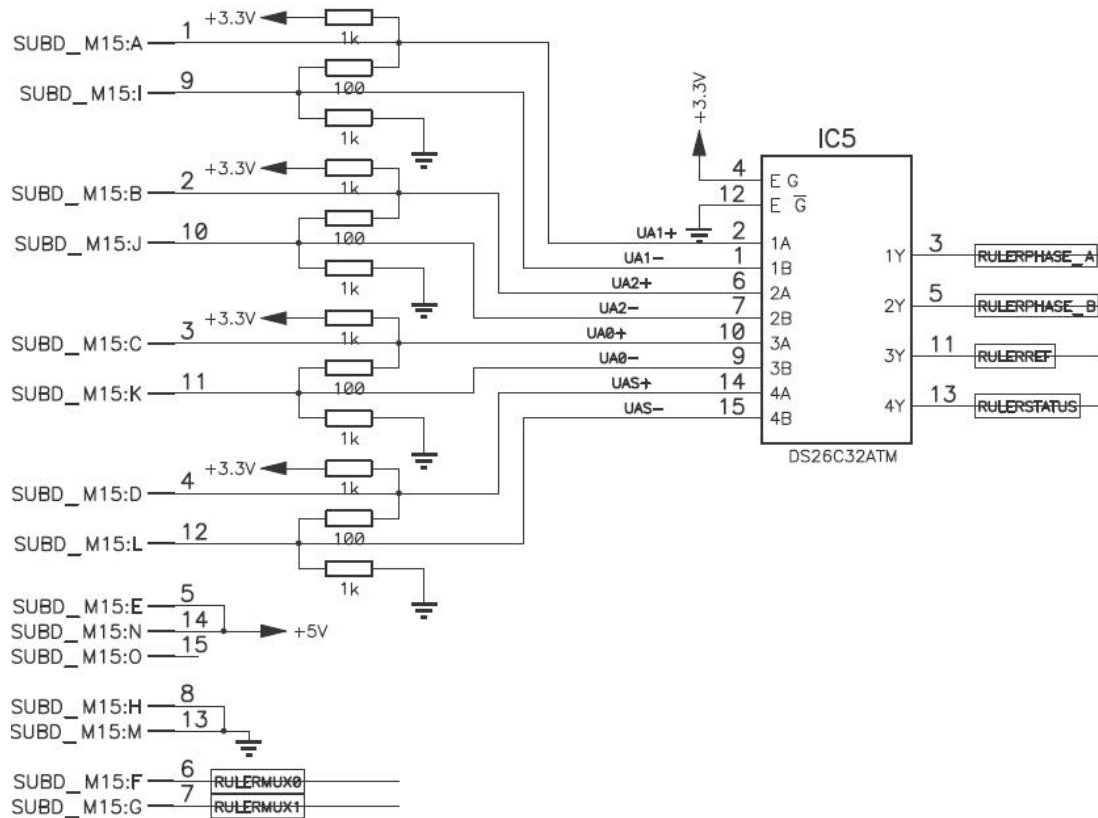


Figure 5.5.: Differential signals from the optical ruler transform to single ended signals.

been added to the prototype card. This is the SRAM which has to store the optical ruler counter data when used in the SPS ring.

The chosen SRAM module is an asynchronous SRAM memory from ISSI⁵, IS61LV25616AL. This memory is organized as 256K words by 16 bits, and it is classified as a CMOS low powered high speed SRAM. The 256K words make an amount of 18 address lines, and 16 parallel data lines. Some buffers are drawn between the data-bus from the slave FPGA to the address lines, these will be enabled if the optical ruler has to be used for addressing the memories (f.ex. by calibration). The transceivers control the flow of data between the data-busses. The flow of data to and from the memory is controlled by the direction and enable inputs of the memory.

⁵Integrated Silicon Solution Inc.

Measurements

As mentioned, the wire scanner has been disassembled, so the optical ruler count data can not be stored. Instead, the active scan signal has been set as an enable signal for the buffer between the function bus and the data bus. Hence the acquired data is actually the motion function data.

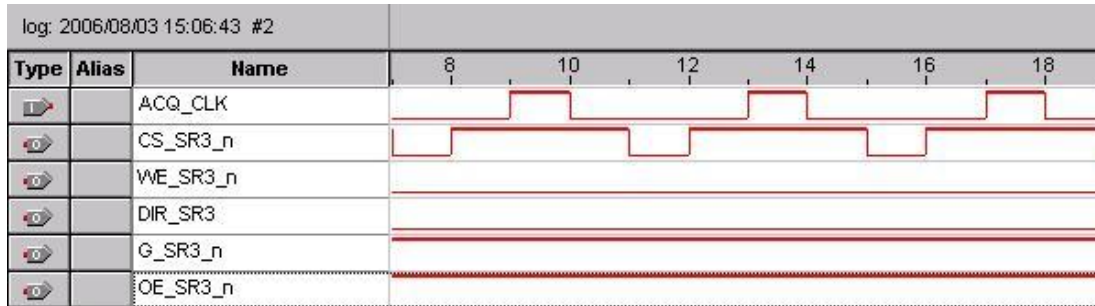


Figure 5.6.: SignalTap verification of the data acquisition at an acquisition frequency of 10 MHz.

In the current test, the function generation clock has been set to 1 MHz, and the acquisition clock is set to 10 MHz. This means that every output value is sampled 10 times. As seen in figure 5.6, the write and chip select signal work as desired. And with the data and new address registered by the rising edge of the acquisition clock, the two-cycles delay is a secure data storage for a 10 MHz acquisition. As the highest acquisition rate of the potentiometer values will be the AD7677's 1 MSps, the SRAM interface is more than fast enough.

5.1.6. Overall test program

To automate the system control and data transfers, a small LabVIEW program has been written. This fills the control register, acquisition clock division register, switch register, FGEN clock division register and scan timeout register. The latter is however not in use, as it has been left out for the time being. When the registers are filled, they are read out afterwards to verify that the correct value has been stored.

By using the LabVIEW program, one does not have to check the correct addresses to manually fill the registers, start a scan and etc. The data read out from the function ROMs or acquisition SRAMs can be displayed in diagrams,

which makes it easier to roughly verify the data than just watching the stored data values.

Measurements

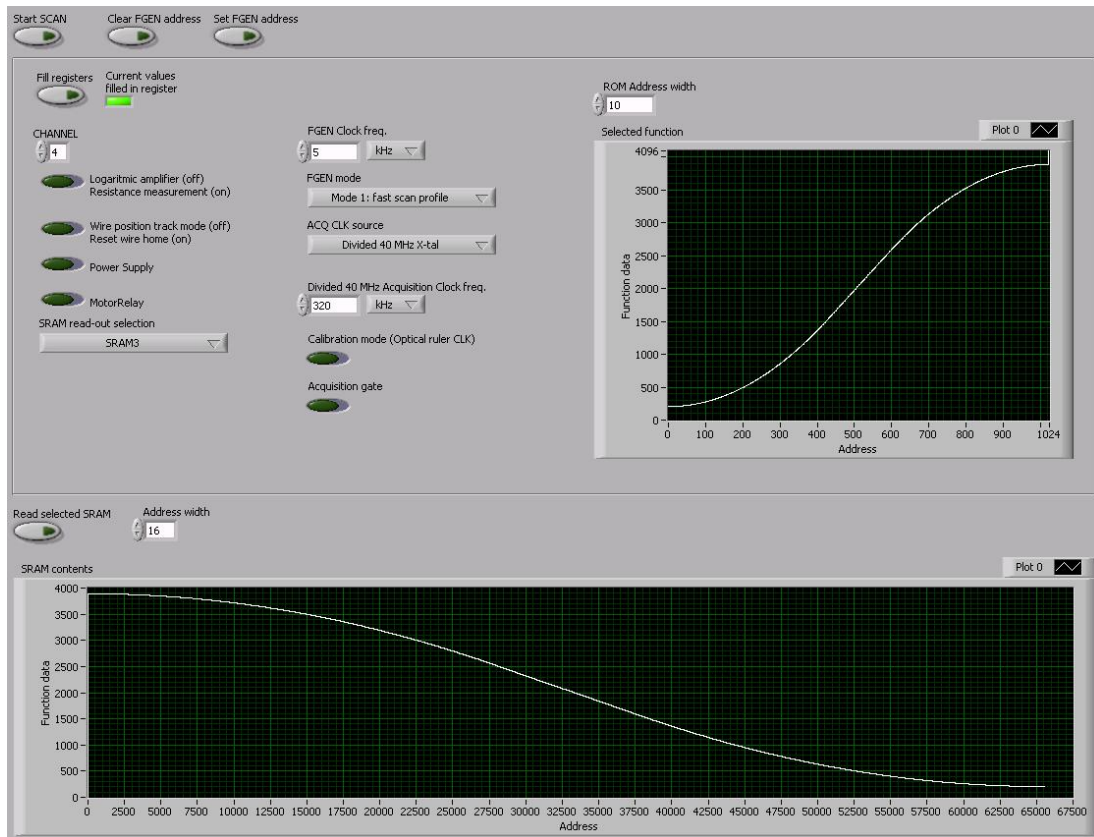


Figure 5.7.: Screen-shot from the LabVIEW program. Function read out after a fast scan profile has been applied. The function data has here been sent for storage in the SRAM3, for simulation purposes as the optical ruler was not available.

By using this program, downloading settings of various registers for testing purposes has become much more efficient and less errors occurred. As shown in figure 5.7, the active function is read out for verification. After a performed scan (out-scan in this case), the acquired data can be read out from the SRAM. Some read values are sometimes different from the written value, but this is due to the fact that the data bus has not been masked when transferring shorter words. Some are already masked in the FPGA, and do not cause problems. However, the data bus connected to the master FPGA has only 8 bits. Hence the upper

eight bits can not be masked by the FPGA when reading from the master FPGA. This data-masking has later been introduced in the Labview program to make sure the read values resemble the correct register values. It has been discussed to add an external buffer to pull the high byte to ground during transfers from the master FPGA, but this will not be implemented due to pin limitations and easy solutions through data-masking in the software.

5.2. Prototype including analog circuits

Due to further progress in the project since the FPGA prototype was done, it was decided to make a new prototype including the analog circuits. Thus further tests could be made, which could now include the entire wire scanner system. While not all the cards were ready, and the measurement setup was restricted, it was still possible to test the control of a rotating wire scanner through the motor driver. Before this, the card had to be debugged to check that the functionality was correct. For the detailed schematics, see appendix I.

5.2.1. VMPS input (ADC0 CH0)

This is a power supply voltage sensor, and should have a level at about 0.6V from the VME J2 connector. This is amplified by 3.3 times and the ADC input should therefore be 1.98 V when the power supply voltage is active. This results in an input voltage of 1.98 V, which is fed to both a comparator and an ADC. The comparator compares this voltage with a divided voltage of $5V \cdot \frac{100}{430} = 1.16V$. During measurement, the comparator is set high if voltages higher than 0.35V and low if lower voltages.

5.2.2. Amplifier voltage check (ADC0 CH1)

To check the motor amplifier voltage input, various DC values were applied. The read out values were then compared to the theoretical values. Some values were out of range and this caused maximum or minimum conversion values, since the range-bit restricted the conversion to voltages from 0 to 2.5V. Otherwise the biggest difference from the expected values was HEX"19", which corresponds to 15mV. This could be explained by the resistor tolerance of 1% in the pre-

V_{in} (J2-10C&11C)	$V_{amp}V_{out}$	V_{adcin}
-1 V	-2 V	3.25!!
-0.6 V	-1.2 V	2.45
0 V	0 V	1.25
0.6 V	1.2 V	0.05
1 V	2 V	-0.750!!
1.5 V	3 V	-1.750!!

Table 5.4.: These are the input voltage levels used during the amplifier voltage check.

ADC expected value	ADC acquired value	Difference (HEX)
FFF	FFF	000
FAD	FA1	00C
7FF	7F6	009
050	06A	019
000	000	000
000	000	000

Table 5.5.: Expected ADC0 conversion values compared to read out values during amplifier voltage check.

amplifying stage. However, at a later stage it was discovered that the ADC0 conversions were influenced by the input impedance, because a resistor had been introduced for the safety of the ADC. Bigger offsets were removed after these resistors were short circuited. To maintain the input safety of the ADC, new operational amplifiers (AD8608) were chosen for the input stage. The operational amplifiers will be fed by a V_{cc} of 5V and a V_{ee} of -300 mV, which keeps the input voltage at the ADC within the safe input range limits.

5.2.3. Amplifier current check (ADC0 CH2)

As for the motor amplifier voltage test-stage, various DC values were applied to the input of the motor amplifier current test-stage. The read out values were compared to the expected values, where the biggest difference showed to be HEX"34" which equals 31 mV. The solution for this is the same as mentioned for the motor amplifier test-stage.

Vin (J2-12C&13C)	Vampioout	Vadcin
-2 V	-1 V	2.25 V
-1.5 V	-0.75 V	2.0 V
-1 V	-0.5 V	1.75 V
0 V	0 V	1.25 V
1 V	0.5 V	0.75 V
1.25 V	0.625 V	0.625 V
1.5 V	0.75 V	0.5 V
2 V	1 V	0.25 V

Table 5.6.: These are the input voltage levels used during the amplifier current check.

ADC expected value	ADC acquired value	Difference (HEX)
E65	E5C	009
CCB	CF1	025
B32	B5A	027
7FF	828	029
4CB	4DC	010
3FF	416	017
332	367	034
198	1CB	032

Table 5.7.: Expected ADC0 conversion values compared to read out values during amplifier current check.

5.2.4. Function generator loop (ADC0 CH3)

The motion function is transformed to an analog signal. To check that this signal is correct, it is fed back to an analog-to-digital converter. Before the ADC, an attenuator inverts the signal and changes the offset to only have positive values from 0 to Vref (2.5V).

A fast scan profile was selected, and the ADC was set to the corresponding channel for the fed back motion function signal. The original function stored in the ROM starts at value 205 and ends on value 3890. By using a loop in Labview to acquire the stored values during a scan, the function shown in figure 5.8 was drawn.

This function looks correct, but the end values have positive offset values and a smaller overall range compared to the original function. The cause of the offset values was found and solved by removing the safety resistors on the input, as for

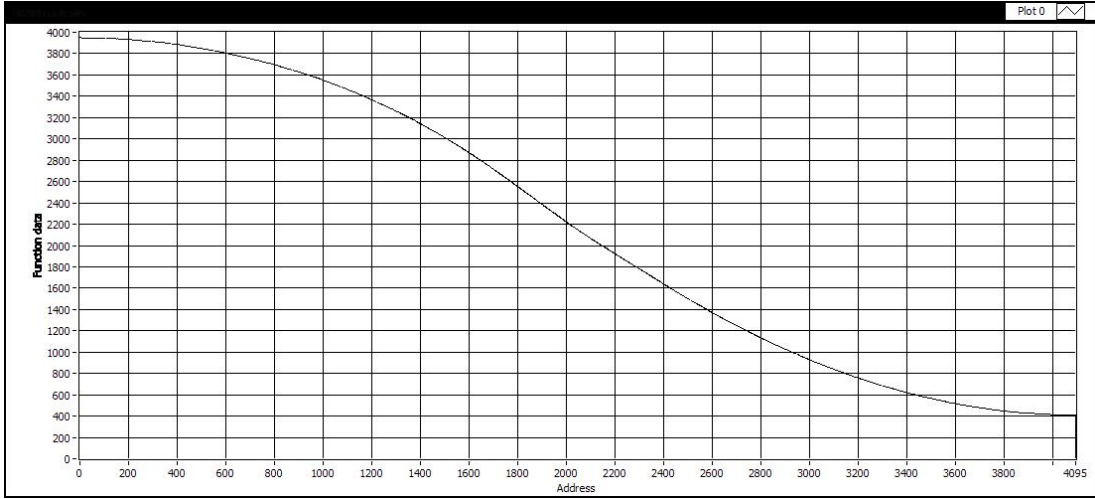


Figure 5.8.: Acquired data at ADC0 CH3 for a fed back fast scan profile. Values were read and displayed graphically in LabVIEW.

the motor amplifier voltage test-stage. Also here, the resistor tolerance introduces an error in the pre-amplifying stage.

5.2.5. Wire temperature circuit (ADC0 CH4 and CH5)

To measure the temperature by means of acquiring the wire-resistance and current, a reference current must first be set. This is solved by using a DAC output voltage with a current generator. The reference current can therefore be set by setting the DAC output voltage. The DAC used has an output range of 5V, from -2.5V to 2.5V. With 12 bits, one LSB then results in $V_{1LSBDAC0} = \frac{5V}{2^{12}} \approx 1.22mV/LSB$.

$$I_{out} = -\frac{R_{98} \cdot V_{in}}{R_{79} \cdot R_{106}} = -\frac{2M\Omega \cdot V_{in}}{1M\Omega \cdot 2k\Omega} = -\frac{V_{in}}{1k\Omega}$$

$$V_{ADC04} = A \cdot (I_{out} \cdot R_L) = -1 \cdot (I_{out} \cdot 2.2k\Omega)$$

The ADC0, which is used to acquire the wire resistance- and current signals, has a 12 bit resolution and as the range bit is set to 0, the range is from 0V to 2.5V. $V_{1LSBADC0} = \frac{2.5V}{2^{12}} = V_{1LSBDAC0} = 610\mu V/LSB$

CH4-wire resistance:

The biggest offset value on the three first measurements is 18 LSBs (see table 5.9), which is an error of 0.44% ($\frac{18LSBs}{2^{12}} \cdot 100\%$). This is however smaller than the 1% tolerance of a resistor and can be neglected. The saturated value of 4.0 V is above the upper range limit and is therefore converted to the maximum value

R_L	DAC01 value	V_{DAC01}	I_{out}	V_{ADC04}
2.2 k Ω	A7F	0.78 V	-0.78 mA	1.72 V
2.2 k Ω	A00	0.625 V	-0.625 mA	1.375 V
2.2 k Ω	981	0.47 V	-0.47 mA	1.034 V
Open circuit	981	0.47 V	-0.47 mA	4.0 V (saturation)

Table 5.8.: Parameters used during the test of the wire temperature circuit. The R_L represents the wire attached to the wire scanner.

Expected ADC04 value	Acquired ADC04 values
B03	AF3 - B02
8CE	8C9 - 8CC
69F	68D - 693
FFF	FFF

Table 5.9.: Expected values compared to read out values at ADC0 CH4, which represents the wire resistance.

as expected.

CH5-wire current:

DAC01 was loaded with value C00, which corresponds to 1.25 V. The I_{out} is then -1.25 mA, and the $V_{ADC05} = A \cdot I_{out} \cdot R_{119} = (-1) \cdot (-1.25mA) \cdot 100 = 0.125V$. This results in an expected ADC value of $\frac{V_{ADC05}}{V_{1LSBADC0}} = \frac{0.125V}{610\mu V} = 204(HEX'0CC')$. The read ADC values range from 0C5 to 0CA, which results in a maximum error of 7 LSBs (0.17 %).

To simulate the loss due to the wire thermionic emission, a leakage current was introduced by pulling a 10k Ω resistance to GND. This reduces the current through the reference resistor to $I_{withleakage} = \frac{10k\Omega \cdot 2.3k\Omega}{2.3k\Omega} = 0.102mA$, which results in an input voltage of $V_{ADC05} = A \cdot I_{out} \cdot R_{119} = (-1) \cdot (-0.102mA) \cdot 100 = 0.102V$. The expected ADC conversion value is then $\frac{V_{ADC05}}{V_{1LSBADC0}} = \frac{0.102V}{610\mu V} = 167(HEX'0A7')$. The read out value was HEX'0AE', which is an error of 7 LSBs (0.17 %).

5.2.6. Logarithmic amplifier acquisition (ADC1)

A sine wave was applied on the input of the logarithmic amplifier. As the amplifier saturated for very small signals, a R262 was removed to reduce the amplification from 2 to 1. For full scale, the signal was adjusted to 492 mVp-p and 238.7 mVoffset while measuring on the output of the logarithmic amplifier. A frequency of 781.25 Hz was selected to acquire exactly two periods during 1024 samples at

a sampling frequency of 400 kHz, as the FFT function in MS Excel only works with a number of samples which has a power of two.

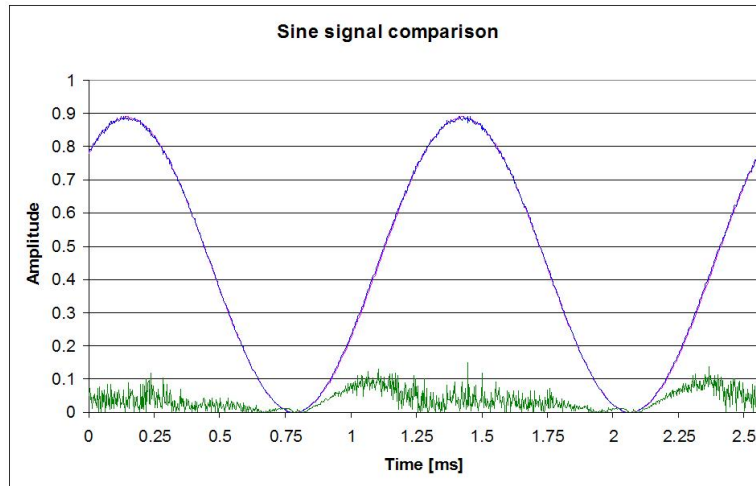


Figure 5.9.: The acquired logarithmic input signal has been linearized and compared to an ideal sine wave of the same amplitude and phase. The green graph shows the difference between the ideal sine wave and the acquired sine wave times ten as the distortion of the captured logarithmic amplified signal.

By linearizing the sampled logarithmic values, the original sine wave can be plotted. To compare the acquired sine wave to an ideal sine wave, a second sine wave with the same amplitude, frequency and phase has been generated using the sine function in Microsoft Excel. This has been done to have a look at the distortion created by the logarithmic transform, which is shown in figure 5.9, where the distortion (times 10) is being displayed in green. As a further study of the distortion, a FFT transform was applied to plot the signal spectrum of the acquired signal (see figure 5.10).

When performing a FFT transform in Microsoft Excel, the absolute values of the FFT transformed values form the amplitude of each indexed frequency. The index ranges from 0 to the sampling frequency, with steps of sampling frequency divided by number of samples. The signal spectrum of the acquired sine signal can then be plotted for the amplitude as a function of the frequency. Figure 5.10 shows that the fundamental harmonic at 781.25 Hz has an amplitude of a factor 128 times (42 dB) the second harmonics at 1562.5 Hz. Thus the higher harmonics are relatively well attenuated, but there is some loss compared to the input signal which attenuates the harmonics by 60 dB (see figure 5.11).

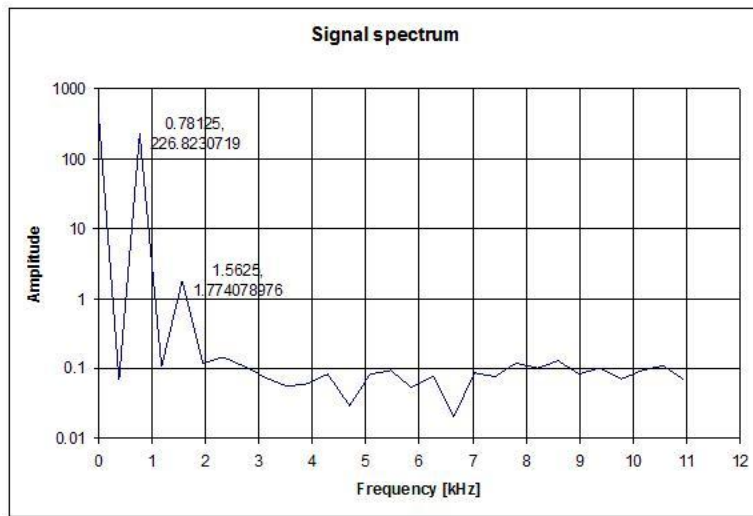


Figure 5.10.: Signal spectrum of the values acquired from the logarithmic amplifier, found by using the linearized values from the logarithmic function in Microsoft Excel.

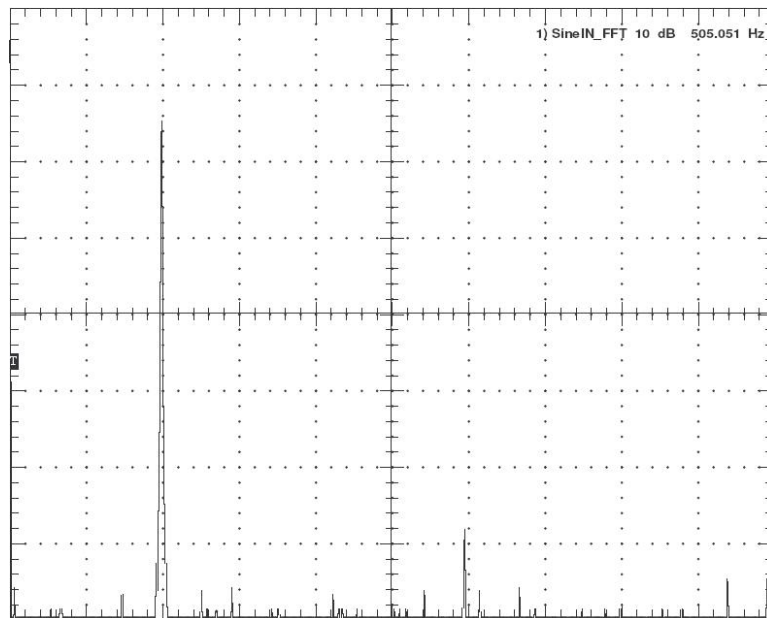


Figure 5.11.: Signal spectrum of the applied 781.25 Hz sine wave input on the logarithmic amplifier, captured by Wavestar software for the oscilloscope.

5.2.7. Potentiometer input with sinusoidal signal (ADC2)

A 100 Hz sine wave of 600 mVpp was applied to the potentiometer channel 1. This was acquired by a 100 kHz sampling frequency, which results in 1000 samples per period.

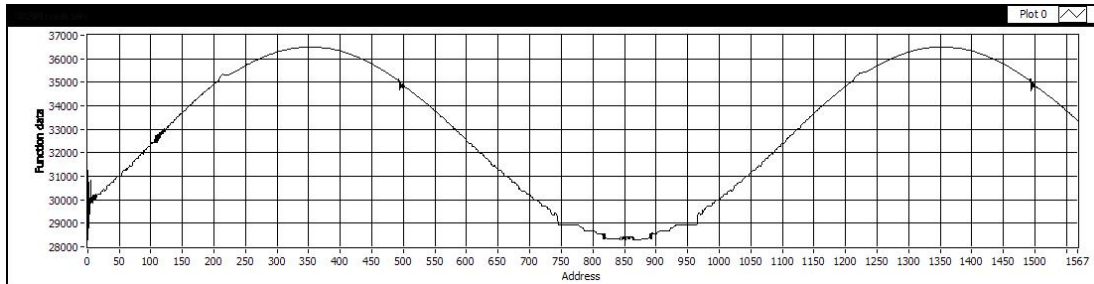


Figure 5.12.: Acquired sine wave at ADC2. Noise occurred due to noise on the reference input.

As seen in figure 5.12, there were however some distortions. The first suspicion was that not all bits were acquired, but the distortion would then probably look different and it was seen on the acquired values that all the bits were toggling. When measuring the reference voltage of 2.5 V, it was observed that there was severe noise at approximately 600 Hz on the reference input. By adding a 47 μF electrolytic capacitor and a 1 μF ceramic capacitor from the reference input to ground, the noise was taken away.

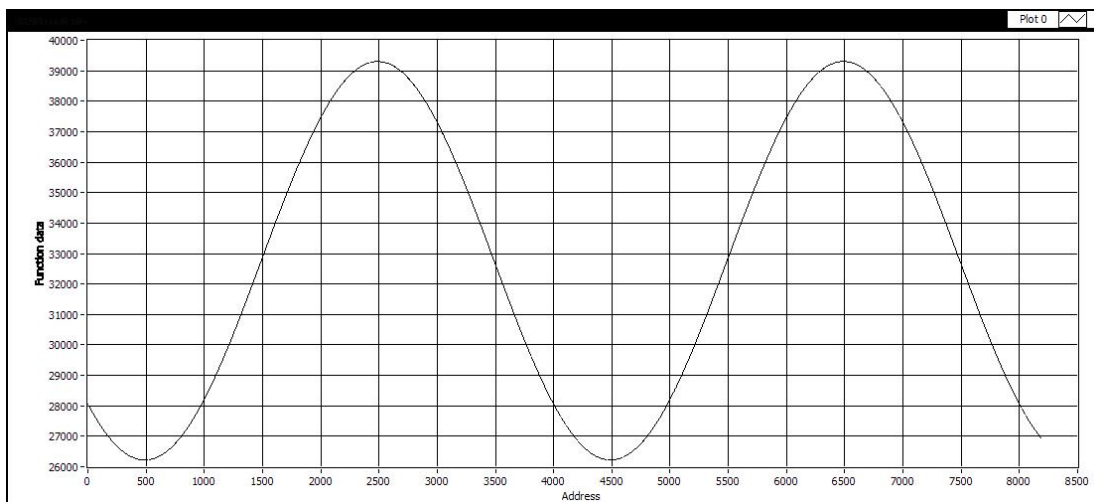


Figure 5.13.: Acquired sine wave at ADC2 after filtering capacitors were added to the reference input.

A signal with the same frequency as before was used, but a bigger amplitude and a much higher sampling frequency of 400 kHz was applied. Thus one period contains 4000 samples, and as seen on figure 5.13, the sine wave is now much smoother. As seen in figure 5.14, the first harmonic of the acquired sine wave is attenuated by about 76 dB. The harmonics are thus very well attenuated.

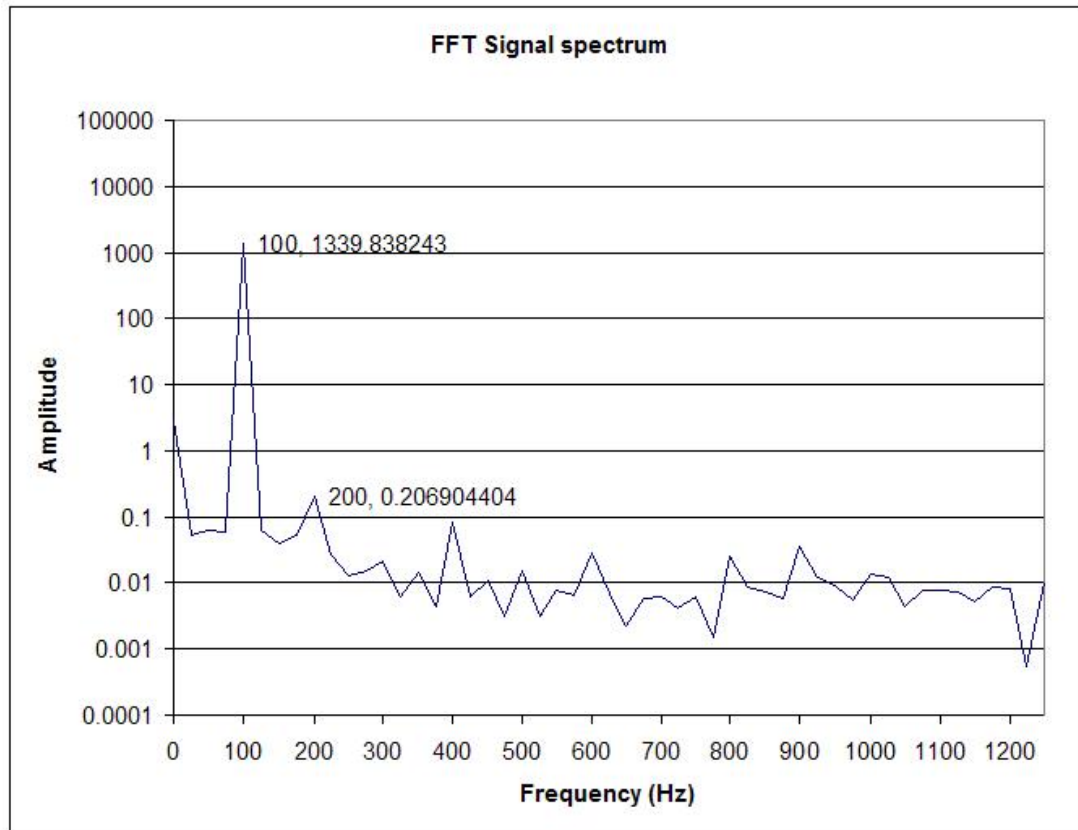


Figure 5.14.: Signal spectrum of the values acquired from the potentiometer input, found by using the values stored in an excel sheet.

5.2.8. Potentiometer accuracy (ADC2)

In order to check the position accuracy of the potentiometer circuit, a potentiometer was connected to channel 1 and acquired for distances of 5 mm. However, due to line capacitance, the power lines with sense feedback were oscillating at roughly 1 MHz, which led to large differences from one acquisition to the next. To solve this, a 3.3 nF capacitor was applied between the output and sense input, thus the bandwidth was decreased to about 2.4 kHz. The power supplies for the

5.2. PROTOTYPE INCLUDING ANALOG CIRCUITS

potentiometer were then improved drastically, as the ripple was insignificantly small.

Potentiometer			
<i>cm</i>	<i>V</i>	<i>ADC RMS</i>	<i>ADC mean</i>
-7.5	2.500	5.120	16442.62
-7.0	2.333	2.771	17537.3
-6.5	2.167	1.981	18631.64
-6.0	2.000	1.616	19712.91
-5.5	1.833	1.339	20760.32
-5.0	1.667	1.159	21879.5
-4.5	1.500	1.069	22961.91
-4.0	1.333	1.001	24059.64
-3.5	1.167	1.045	25128.48
-3.0	1.000	1.013	26220.91
-2.5	0.833	0.989	27316.75
-2.0	0.667	0.962	28419.85
-1.5	0.500	0.943	29489.02
-1.0	0.333	0.960	30532.5
-0.5	0.167	0.870	31670.25
0.0	0.000	0.876	32760.18
0.5	-0.167	0.910	33839.67
1.0	-0.333	0.842	34922.67
1.5	-0.500	0.843	36019.79
2.0	-0.667	0.800	37093.61
2.5	-0.833	0.849	38192.04
3.0	-1.000	0.871	39240
3.5	-1.167	0.838	40358.05
4.0	-1.333	0.881	41441.88
4.5	-1.500	0.924	42530.78
5.0	-1.667	0.976	43600.35
5.5	-1.833	1.101	44721.03
6.0	-2.000	1.292	45812.91
6.5	-2.167	1.712	46877.57
7.0	-2.333	2.567	47947.8
7.5	-2.500	5.424	49063.11

Table 5.10.: The potentiometer was set to the correct position by measuring the corresponding voltage. By sampling 1024 values, the RMS and mean values of the ADC2 conversion could be found.

The positions were set using the corresponding voltage levels, as shown in table

5.10. As it is crucial to keep invariance in mid-position (0 cm), it is desired to study the standard deviation. The standard deviation is a measure of how widely values disperse from the mean value. To be capable of estimating the standard deviation, several samples are needed. Thus every position was sampled 1024 (2^{10}) times. The dispersed values are presented in LSBs as the ADC RMS in table 5.10.

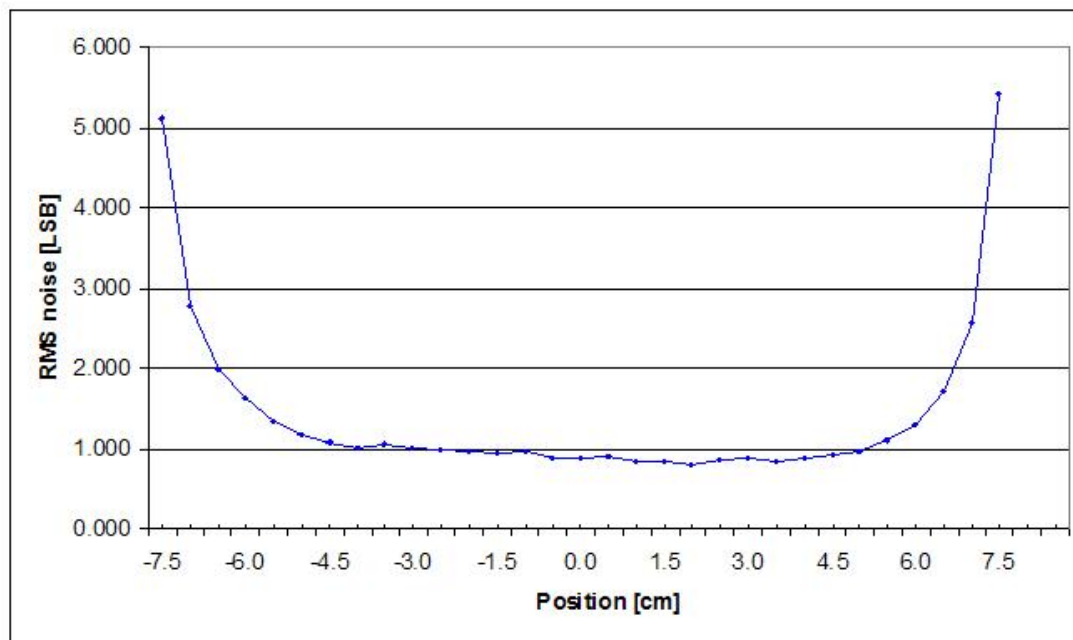


Figure 5.15.: Accuracy measures of the potentiometer. The displayed data represents the RMS dispersion level from the mean value over 1024 samples. It is important that this is at a minimum in the mid-position, where the beam profile measurements will be made.

As shown in figure 5.15, the dispersion is less than 1 LSB in the mid-positions. This means that there is less than 1 LSB position error ($1 \text{ LSB} \hat{=} 4.58 \mu\text{m}$).

5.2.9. Wire relay circuit

The wire relay circuits consist of switching transistors that remotely control 4 relays. The expected output levels were checked and found to be correct.

5.2.10. Scope output signals

The "scope" output is a multiplexed output, with a hexadecimal switch to select which channel to direct to the output on the front panel. This output is applied to a Lemo connector in order to easily connect it to an external oscilloscope input. This output allows checking vital analogue signals without dismounting the VME-crate. The multiplexing of all signals has been checked and found to be ok.

5.2.11. LabVIEW testbench for WSMCC

In total, 15 wire scanner motion control cards will be produced. In order to test these efficiently, it is desired to have an automatic testbench. A small testbench has been made in LabVIEW for this purpose, yet it is more an functional test rather than complex circuit tester. Not all circuits have been implemented, but the most essential FPGA and ADC functions are. To use this LabVIEW program, the CAEN USB-VME bridge must be used, the WSMCC has to be set to physical address HEX"0" and DC values have to be applied to certain channel inputs. Since some of the inputs on the J2-VME connector are grounded, there should not be anything connected to the backplane side of the J2-VME connector.

The test flow has been set as shown in figure 5.16. The FPGA functions are checked first, then the ADC conversions. To apply the DC values used to verify the ADC conversions, a tiny card containing a resistor voltage dividing network and voltage followers has been attached to an VME extension card. The schematics for this card can be seen in figure 5.17.

When these DC values are applied to the correct inputs, the LabVIEW program can be initiated. The LabVIEW front panel will display percentage results of the difference between the expected value and measured value.

The details of the flowcharts and LabVIEW block diagrams can be found in appendix G.

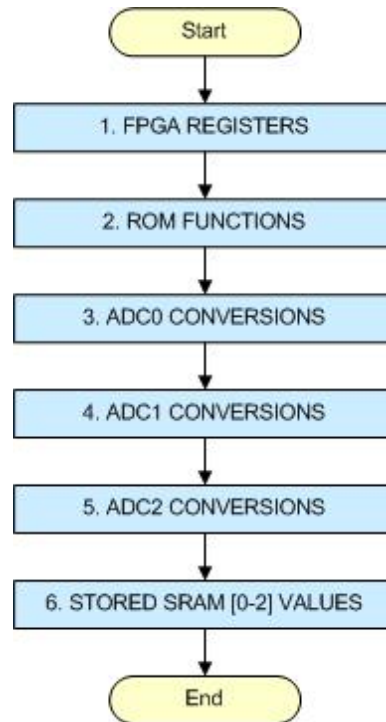


Figure 5.16.: Test flow from start to end for the LabVIEW testbench.

GND	-2V	-1V	-0.5V
J2-2A	J2-1A	J2-1C	J2-10C
J2-2C	J2-13C	J2-18A	J2-14A
J2-11C	J2-18C	-	-
J2-12C	-	-	-
J2-14A	-	-	-
J2-17A	-	-	-
J2-17C	-	-	-

Table 5.11.: In order to use the LabVIEW testbench, the DC values must be connected to the pins shown in this table.

5.2.12. Controlling a wire scanner

A rotating wire scanner was mounted and connected to the system. The aim of this test was to control a wire scanner by the motion control card through the power amplifier on the VME-J2 connector. The system then consisted of a wire scanner motion control card, a power amplifier and a wire scanner with its motor and potentiometer. An USB-VME bridge introduced the communication interface for the remote control.

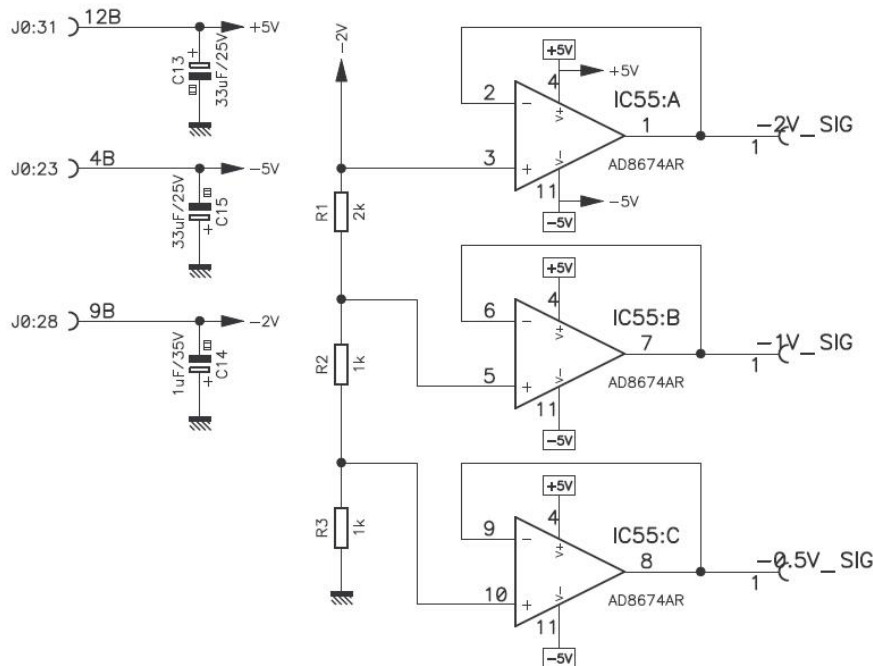


Figure 5.17.: Schematics for the small card providing the DC values for the ADC inputs. This card is attached to the VME extension card, where the signals can easily access the J2-VME connector by pulling short wires.

The motor position was manually set to the initial position, and the control voltage was checked to be correct. The power amplifier for the motor driver was activated. The motor was expected to stand still initially, but it was oscillating and kept rotating at a very high speed. Closer study showed that it was moving in the wrong sense, and thus kept oscillating around the zero-point in the dead-zone of the rotational potentiometer. This meant that the motor polarization was inverted.

The polarization was changed and the amplification of the motor driver was reduced to decrease the risk of oscillation. At the second test the motor control worked correctly, but the lag between the function generator voltage and motion feedback voltage was big due to the low amplification in the closed control loop. By increasing this amplification close to the point of oscillation, the lag time was reduced drastically, yet it still remains to find the fully optimized PID regulation values.

By using the measurements shown in figure 5.18, we can estimate the maximum wire speed achieved. A wire supporting arm for the rotational wire scanner has

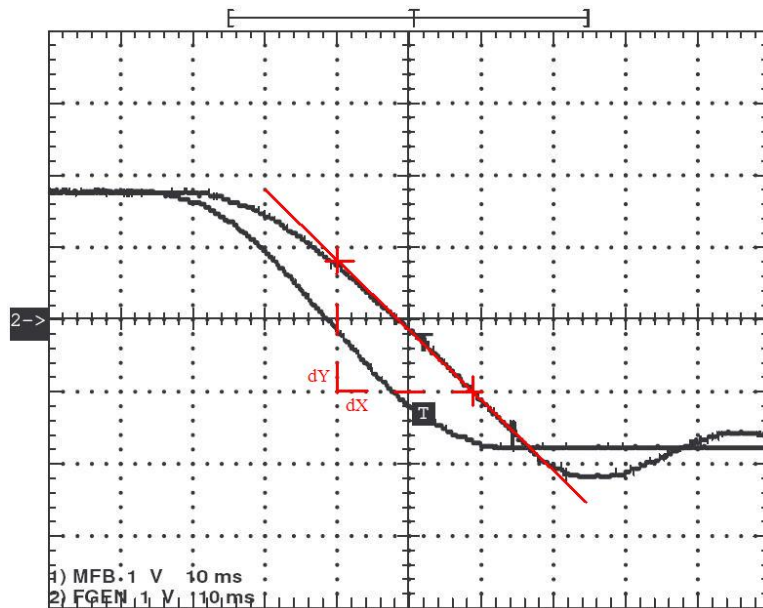


Figure 5.18.: When stepping through the motion function with a high speed, the actual motion of the wire scanner lags behind. The feedback loop tries to compensate the lag, and it therefore overshoots by the end of the stroke.

a length of $r = 20\text{cm}$. If a motion of 345 degrees resembles the full stroke, the distance of the stroke is:

$$d = \frac{345}{360} \cdot 2\pi \cdot r = \frac{345}{360} \cdot 2\pi \cdot 0.2\text{m} = 1.2\text{m}$$

From figure 5.18 we find the derivative of the linear part to be $dY = 1.8\text{V} = \frac{1.8\text{V}}{5\text{V}} \cdot 1.2\text{m} = 0.432\text{m}$ and $dX = 19\text{ms}$. From this we can find the speed in the linear part, which is:

$$v = \frac{dY}{dX} = \frac{0.432\text{m}}{19\text{ms}} = 22.7\text{m/s}$$

Which is more than twice the required speed of 10 m/s for the rotational wire scanner. The specifications are met, however the measurements were done without the wire arm, which eventually will contribute to the inertia of the system and reduce the maximum achievable speed.

6. Conclusion

The aim of this project was to develop the wire scanner motion control card in order to increase the acquisition accuracy of the wire position and add advanced programmable functionality.

The measurements on the most recent prototype including the analogue circuits have shown high accuracy of the acquired position data and precise motor motion-control with programmable features.

ADC-, DAC- and SRAM-control has been successfully implemented in the FPGA. The acquisition control is triggered by the handshake of the ADCs and the acquired data are stored reliably in the SRAM with a very high maximum rate of 20 MBPS, which is much higher than the ADC conversion rate.

The VME-bus communication interface works very well and the newly implemented block transfer mode has improved the memory read-out by two orders of magnitude. By optimizing the data acknowledgement delay, the transfer rate has been even further improved than at earlier measurements. One SRAM of 512 kB can now be read-out in 0.1 s, which results in a transfer rate of over 5 MBPS and this is well within the requirements.

This shows that the design of the wire scanner motion control card with its firmware has fulfilled the foreseen specifications successfully.

Improvements

The use of three FPGAs has introduced multiple register flags and debugging restrictions which could be avoided by using one BGA package instead. Also the debugging would be simplified if the full databus width of 16 bits was added to the master FGPA, where the most critical control signals can trigger the logic analyzer.

The accuracy of the position acquisitions could be further improved if the preamplifier was adjusted to use the entire input range of the differential ADC.

7. Abbreviations, list of figures and list of tables

Abbreviations

ALICE : A Large Ion Collider Experiment

AS : Active Serial

ASMI : Active Serial Memory Interface

ATLAS : A Torodial LHC ApparatuS

CERN : Conseil Européen pour la Recherche Nucléaire

CMS : Compact Muon Solenoid

BGA : Ball Grid Array

BGI : Beam Gas Ionization

BLT : BLock Transfer

BSR : Beam Synchrotron Radiation

DAC : Digital-to-Analog Converter

EMI : Electromagnetic Magnetic Interference

FP : Flat Pack

FPGA : Field Programmable Gate Array

GeV : Giga electron Volt

IC : Integrated Circuit

ISSI : Integrated Silicon Solution Inc.

JIC : JTAG Indirect Configuration

JTAG : Joint Test Action Group

LED : Light Emitting Diode

LSB : Least Significant Bit
LUT : Look-Up-Table
MIF : Memory Initialization File
MSB : Most Significant Bit
LEP : Large Electron-Positron collider
LHC : Large Hadron Collider
PCB : Printed Circuit Board
PID : Proportional-Integration-Differentiation
PMT : Photo Multiplier Tube
POR : Power-On-Reset
PQFP : Plastic Quad Flat Pack
PS(1) : Passive Serial
PS(2) : Proton Synchrotron
ROM : Read Only Memory
RTL : Register Transfer Level
SAR : Successive-Approximation Register
SEM : Secondary Emission current
SOF : SRAM Object File
SOP : Small Outline Package
SPS : Super Proton Synchrotron
SR : Set-Reset
SRAM : Static Random Accessable Memory
TeV : Tera electron Volt
TQFP : Thin Quad Flat Pack
VHDL : Very-High-Speed-Integrated-Circuits Hardware Description Language
VHSIC : Very High Speed Integrated Citrcuits

VME_x : VERSA-Module Eurocard eXtended bus
WSMCC : Wire Scanner Motion Control Card
WWW : World Wide Web

List of Figures

1.1.	An illustration of the path of the beams injected into LHC (red and green arrows).	3
1.2.	An overview of the experiments in the LHC.	3
1.3.	Example of a: Top; Phase space diagram of the beam intensity distribution. The axis are defined as particle position μ and particle angle with respect to the nominal trajectory. Bottom; Projection of the intensity distribution onto the position coordinate.	5
1.4.	Photo and schematic of operation of a flying wire scanner.	6
1.5.	The profile measurement setup using a wire scanner. The scintillator and the photo multiplier tube detects the charged particles of the shower, and the ampere-meter measures the SEM signal.	7
2.1.	Entire wire scanner system.	9
3.1.	An overview of the wire scanner motion control card. A complete functional diagram can be found in appendix A.	13
3.2.	Wirescanner position control.	14
3.3.	Wire scanner movement functions: a) Linear offset mode; b) Accelerated and decelerated fast scan mode; c) Linear slow scan mode. The exact profiles will be calculated in section 4.5.2	15
3.4.	Wire scanner position acquisition diagram.	16
3.5.	Optical Ruler sine-wave signals fed to and digitized by the EXE 610C.	18
3.6.	Digitization circuit phase train signals.	19
3.7.	Basic timing diagram for the AD7677.	20
3.8.	Basic timing diagram for the AD7484: Top; Read cycle timing. Bottom; Write cycle timing.	22
3.9.	Basic timing diagram for the AD7938 in parallel word read mode.	23
3.10.	Block diagram of the SRAM.	25
3.11.	Read cycle, controlled by chip enable.	26
3.12.	Write cycle, controlled by chip enable.	26
3.13.	VME-bus timing diagram. The edges of DSx depend on the AS edges.	27

4.1. FPGA and Serial configuration device programming in AS mode. Figure origins from figure 4-2 in reference [15]	32
4.2. JTAG Configuration of the FPGA and indirect Serial Configuration device programming through the Serial Flash Loader. Figure origins from figure 4-25 in reference [15]	33
4.3. Principle of the Serial Flash Loader interface bridge compared to the conventional configuration scheme. Figure origins from figure 1 in reference [16]	34
4.4. Configuration setup for multiple devices using the JTAG connector and a serial configuration device.	35
4.5. Block overview of the master FPGA system.	37
4.6. Block symbol for the VME address verifier. Source code can be found in appendix B	40
4.7. ADC signal flow solution based on the AD7938 timing in read mode.	41
4.8. SRAM write signal flow control.	42
4.9. Block overview of the signals connected to the first slave FPGA. .	43
4.10. Block symbol for the control unit of the first slave. Most output signals are VME-transferred values stored in the registers.	44
4.11. Block schematic of the VHDL top entity of the function generation.	47
4.12. Fast scan set value function.	49
4.13. Linear offset set value function as derived in matlab. Y-scale adjusted in accordance with a linear wire scanner (130mm stroke).	50
4.14. Fast scan set value function, speed and acceleration plotted in matlab. Y-scale adjusted in accordance with a linear wire scanner (130mm stroke).	51
4.15. Linear slow scan set value function plotted in matlab. Y-scale adjusted in accordance with a linear wire scanner (130mm stroke).	51
4.16. An extract from the generated .mif-file for the fast scan profile, which has been printed while looping through the matlab generated vectors. The .mif-file is a standard used by Altera, where data-width and address-depth must be defined. The contents are listed with the address first, followed by the corresponding data.	52
4.17. Block schematic of the function check entities.	52
4.18. Block symbol for the optical ruler quad decoder	53
4.19. Optical ruler quad decoder state machine.	55
4.20. Conditions to change state in the optical ruler quad decoder state machine.	56
4.21. Block overview of the signals connected to the second slave FPGA.	57
4.22. Block symbol for the control unit in slave two.	58
4.23. Holding registered errors until the asynchronous reset is activated.	58
4.24. State machine controlling the seven segment LED display.	59

5.1. Measurement setup for the FPGA prototype. As the wire scanner is currently disassembled and some of the collaborating cards are not available, simulation values were fed to the memories. The FPGA prototype aims to check the programming hardware, memory storage and VME transfers.	62
5.2. Configuration scheme for the two FPGAs.	63
5.3. Low voltage drop-out regulator creating a 1.2V power supply from a 3.3V voltage source.	68
5.4. SubD15 male connector.	68
5.5. Differential signals from the optical ruler transform to single ended signals.	70
5.6. SignalTap verification of the data acquisition at an acquisition frequency of 10 MHz.	71
5.7. Screen-shot from the LabVIEW program. Function read out after a fast scan profile has been applied. The function data has here been sent for storage in the SRAM3, for simulation purposes as the optical ruler was not available.	72
5.8. Acquired data at ADC0 CH3 for a fed back fast scan profile. Values were read and displayed graphically in LabVIEW.	76
5.9. The acquired logarithmic input signal has been linearized and compared to an ideal sine wave of the same amplitude and phase. The green graph shows the difference between the ideal sine wave and the acquired sine wave times ten as the distortion of the captured logarithmic amplified signal.	78
5.10. Signal spectrum of the values acquired from the logarithmic amplifier, found by using the linearized values from the logarithmic function in Microsoft Excel.	79
5.11. Signal spectrum of the applied 781.25 Hz sine wave input on the logarithmic amplifier, captured by Wavestar software for the oscilloscope.	79
5.12. Acquired sine wave at ADC2. Noise occurred due to noise on the reference input.	80
5.13. Acquired sine wave at ADC2 after filtering capacitors were added to the reference input.	80
5.14. Signal spectrum of the values acquired from the potentiometer input, found by using the values stored in an excel sheet.	81
5.15. Accuracy measures of the potentiometer. The displayed data represents the RMS dispersion level from the mean value over 1024 samples. It is important that this is at a minimum in the mid-position, where the beam profile measurements will be made.	83
5.16. Test flow from start to end for the LabVIEW testbench.	85

5.17. Schematics for the small card providing the DC values for the ADC inputs. This card is attached to the VME extension card, where the signals can easily access the J2-VME connector by pulling short wires.	86
5.18. When stepping through the motion function with a high speed, the actual motion of the wire scanner lags behind. The feedback loop tries to compensate the lag, and it therefore overshoots by the end of the stroke.	87
G.1. A flowchart which shows the idea of how to check the FPGA registers using a loop and an address array.	123
G.2. From flowchart to LabVIEW block diagram.	124
G.3. A flowchart which shows the idea of how to use loops and initial datasheet values to confirm that the functions stored in the FPGA ROM is correct.	125
G.4. From flowchart to LabVIEW block diagram.	126
G.5. Flowchart shows the idea for how to check the ADC0 conversions, using a loop to check several channels sequentially.	127
G.6. From flowchart to LabVIEW block diagram..	128
G.7. Flowchart shows the idea for how to check the ADC1 conversion.	129
G.8. From flowchart to LabVIEW block diagram..	130
G.9. Flowchart shows the idea for how to check the ADC2 conversion.	131
G.10. From flowchart to LabVIEW block diagram..	132
G.11. This flowchart shows the idea for how perform a scan and read out the acquired values for diagnosis and graphical display.	133
G.12. From flowchart to LabVIEW block diagram..	134

List of Tables

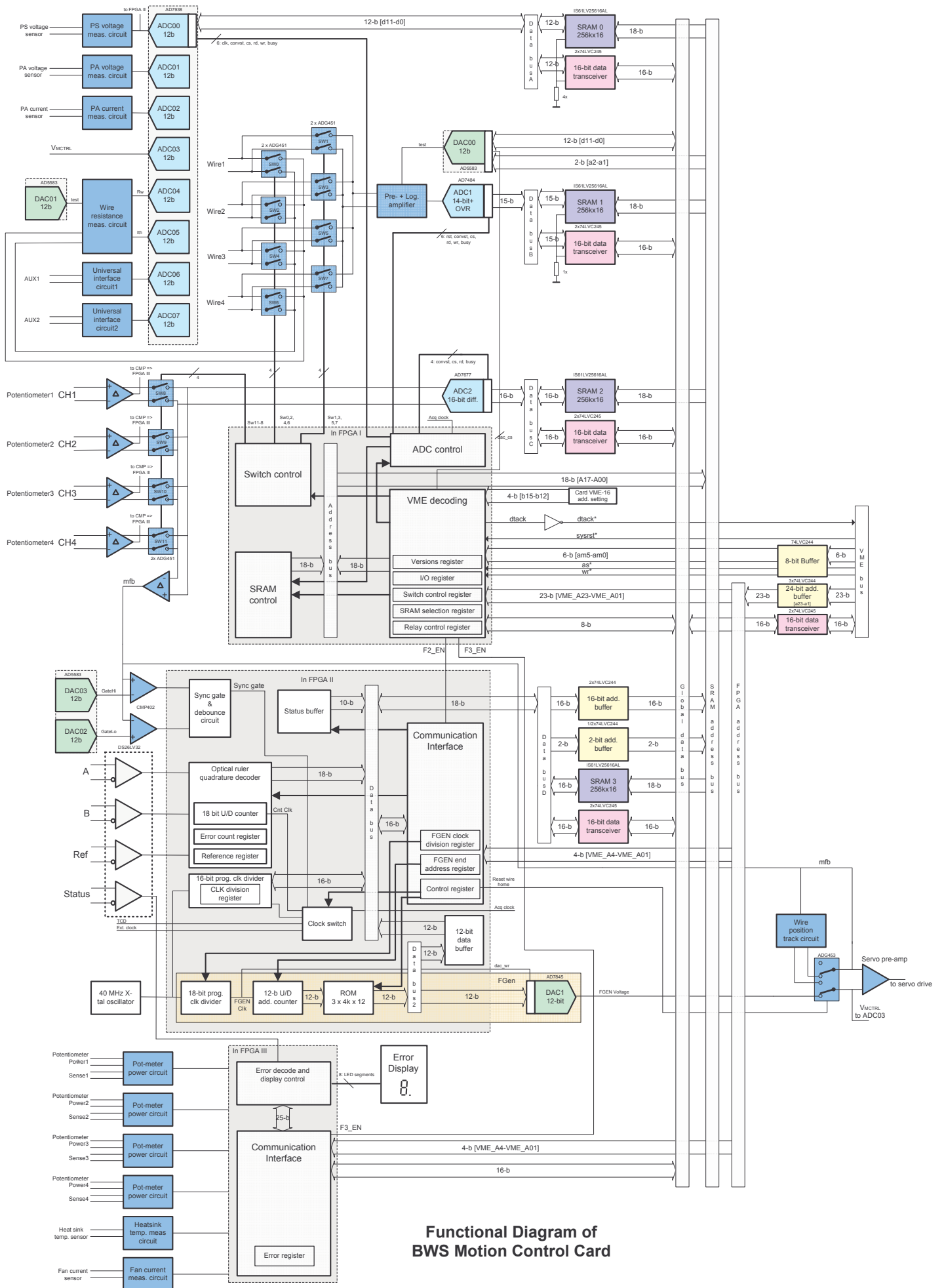
3.1. ADC control register	24
3.2. ADC shadow register	24
4.1. Altera Cyclone II FPGA data	31
4.2. Configuration modes MSEL settings	36
4.3. VME signals for basic operation	38
4.4. Rough VME mapping	39
4.5. Memory mapping of slave one	45
4.6. Mode control reg	46
4.7. Profile selection	48
4.8. Optical ruler phase states	54
4.9. Error messages	60
4.10. Seven segment display codes	61
5.1. Prototype VME signals	65
5.2. Prototype VME power supply signals	67
5.3. Pinning specification	69
5.4. Input test voltages for the amplifier voltage check.	74
5.5. Expected ADC0 conversion values compared to read out values during amplifier voltage check.	74
5.6. Input test voltages for the amplifier current check.	75
5.7. Expected ADC0 conversion values compared to read out values during amplifier current check.	75
5.8. Parameters set to test the wire temperature circuit.	77
5.9. Expected values compared to read out values at ADC0 CH4, which represents the wire resistance.	77
5.10. Potentiometer accuracy measures.	82
5.11. Input values on J2-connector for testbench.	85

Bibliography

- [1] general information about CERN, 2005
- [2] internal Document Server of the BLM-section, 2005
- [3] A. Di Girolamo, *"Studies on the performances of the monitored drift tubes of the Atlas detector"*, 2004
- [4] J. Bosser, J. Camas, L. Evans, G. Ferioli, R. Hopkins, J. Mann and O. Olsen, *"Transverse emittance measurement with a rapid wire scanner"*, 1984
- [5] P. Elmfors, A. Fasso, M. Huhtinen, M. Lindroos, J. Olsfors and U. Raich, *"Wire scanners in low energy accelerators"*, 1997
- [6] J. Bosser and C. Bovet, *"Wire scanners for LHC"*, 1997
- [7] S. Burger, C. Carli, K. Priestnall and U. Raich, *"The PS booster fast wire scanner"*, 2003
- [8] B. Dehning and F. Roncarolo, *"Transverse emittance blow-up due to the operation of wire scanners, analytical predictions and measurements"*, 2005
- [9] VITA VMEbus: *"VMEbus technology"*,
<http://www.vita.com>, 2005
- [10] Stefan Sjöholm and Lennart Lindh *"VHDL for designers"*,
1997
- [11] L. Ponce, R. Jung and F. Méot, *"LHC proton beam diagnostics using synchrotron radiation"*, 2004
- [12] S. Turner, *"CERN Accelerator School - Fifth general accelerator physics course. Vol I"*, 1994
- [13] W. Herr and B. Muratori, *"Concept of luminosity"*, 2005
- [14] Altera, *"Cyclone II Device Handbook, Volume 1"*, 2005
- [15] Altera, *"Configuration Handbook, Volume 1"*, 2005
- [16] Altera, *"Using the Serial FlashLoader With the Quartus II Software"*, 2004

8. Appendix

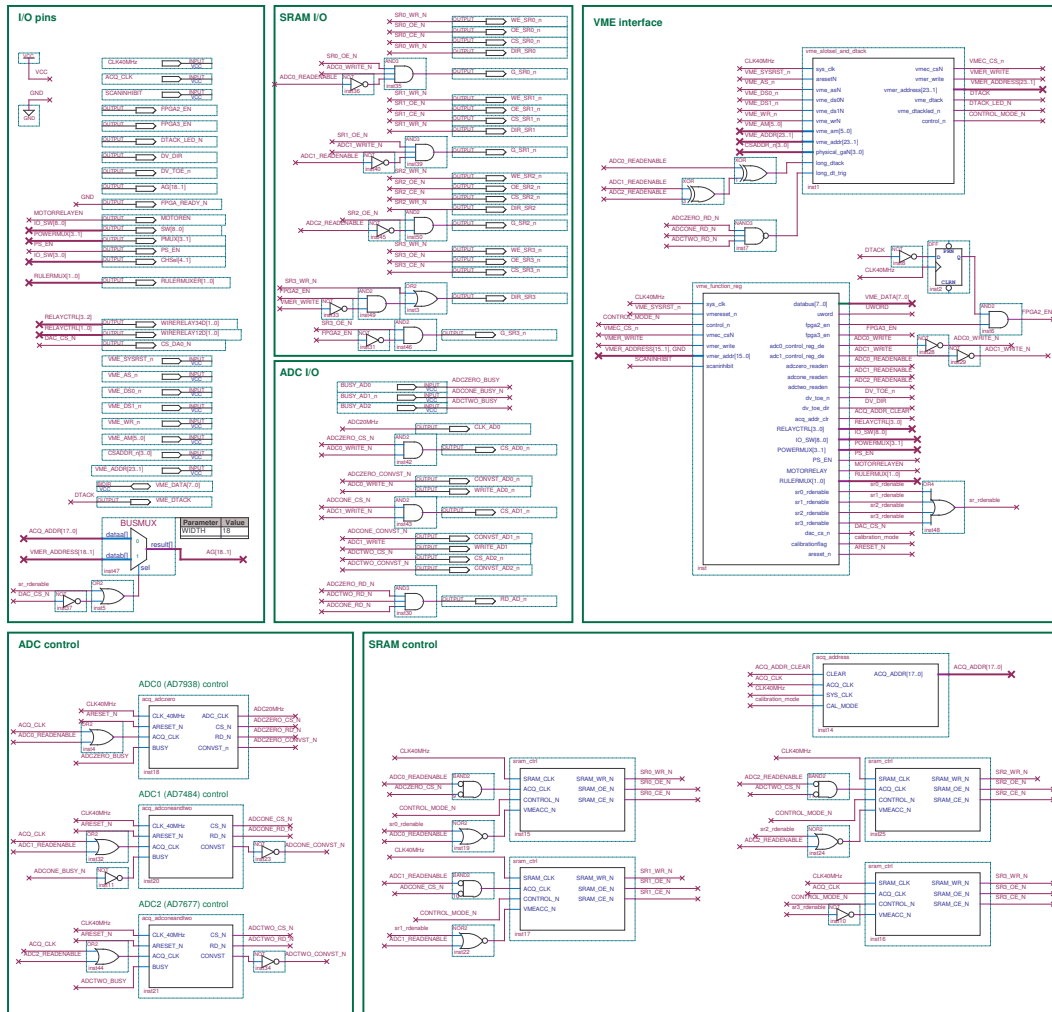
A. WSMCC functional diagram



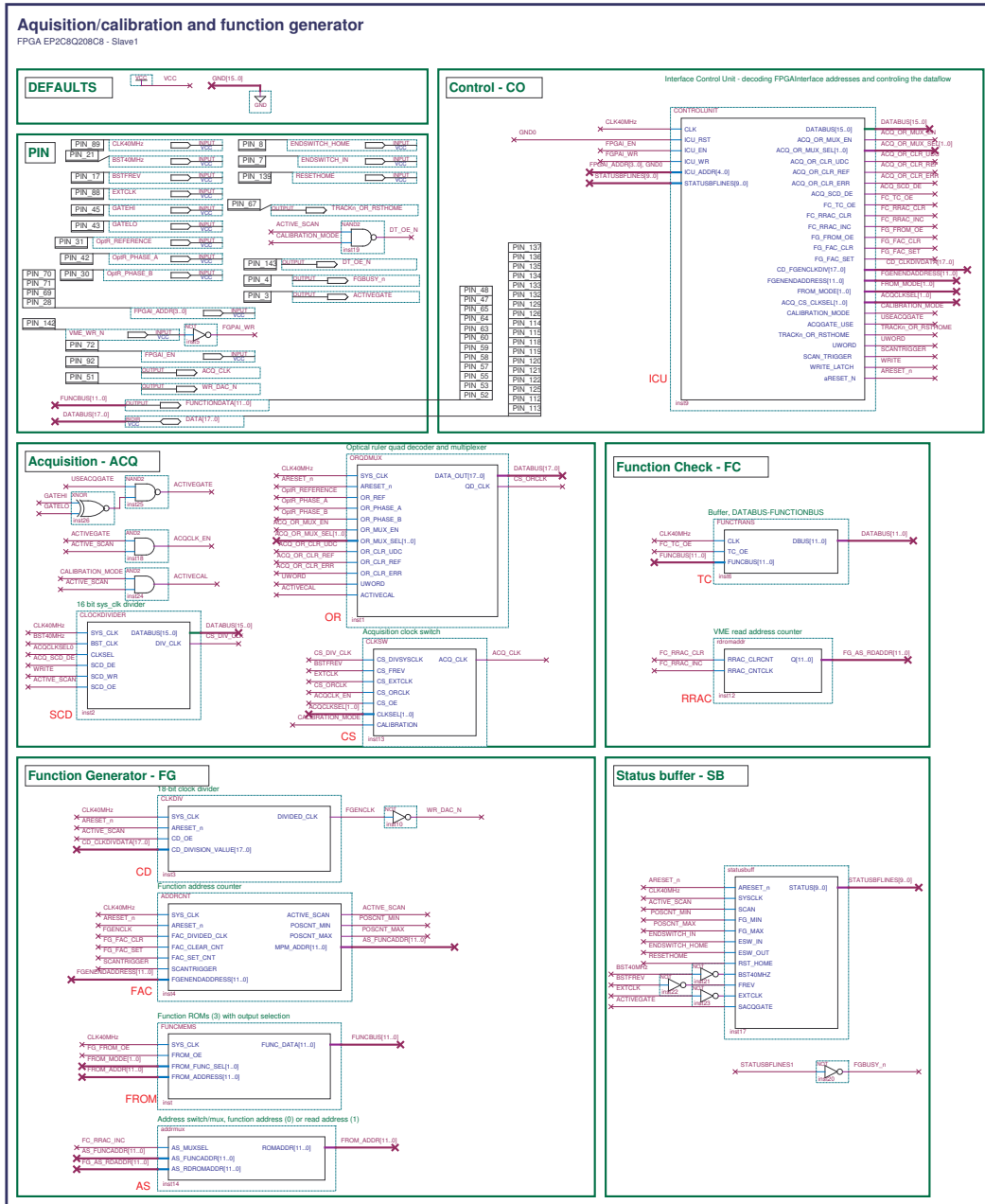
Functional Diagram of BWS Motion Control Card

B. VHDL top level block diagrams

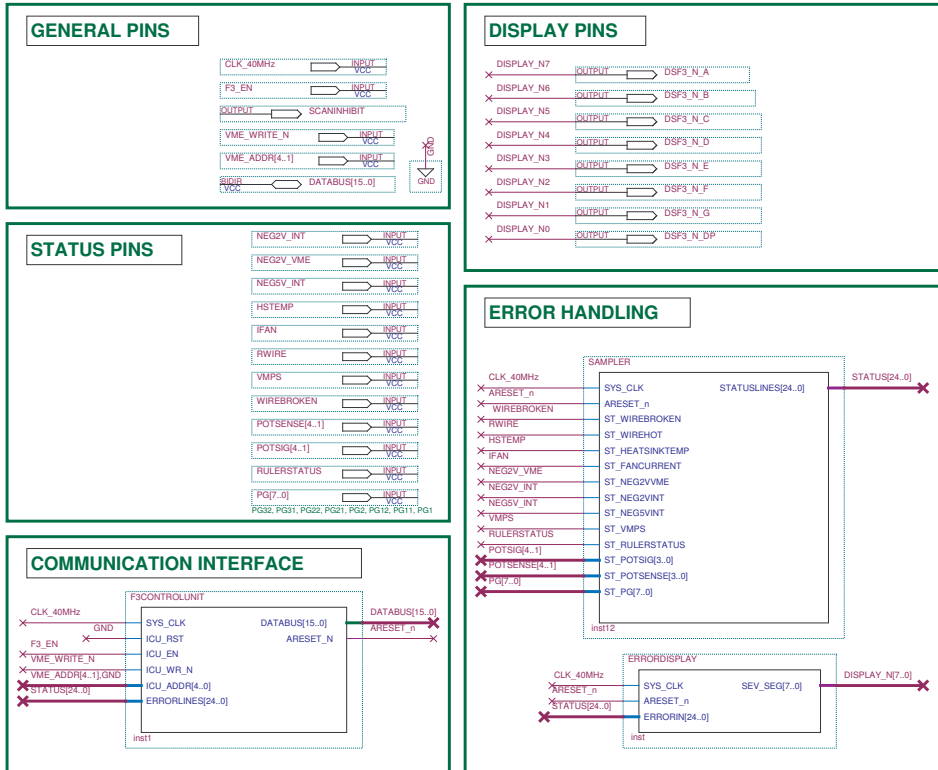
B.1. Master FPGA top level



B.2. Slave1 FPGA top level



B.3. Slave2 FPGA top level



C. FPGA Registers

Register description:

General abbreviations:

R: Read, W: Write, LB: Low Byte, HB: High Byte, LW: Low Word, HW: High Word, HEX: Hexadecimal, DEC: Decimal, X: Don't care (bit not used), D#: Data bit #, NU: Not used.

Registers inside FPGA 1 (VME Data width = 8):

Versions register R only (Address: HEX"4C"/DEC"76"):

D7	D6	D5	D4	D3	D2	D1	D0
BV3	BV2	BV1	BV0	FV3	FV2	FV1	FV0

D3-D0: FPGA version number (hex).

D7-D4: Board version number (hex).

FPGA I/O register R/W (Address: HEX"44"/DEC"68" (LB), HEX"46"/DEC"70" (HB)):

D7	D6	D5	D4	D3	D2	D1	D0
d7	d6	d5	d4	d3	d2	d1	d0
d15	d14	d13	d12	d11	d10	d9	d8

FPGA Switch control register R/W (Address: HEX"48"/DEC"72"):

D7	D6	D5	D4	D3	D2	D1	D0
d7	d6	d5	d4	X	d2	d1	d0

D2-D0: Channel select:

000: SEM1: Sw1 on, Sw2 off, Sw9 on, PowerMux1 off, PowerMux2 off, PowerMux3 X.

001: SEM2: Sw3 on, Sw0 off, Sw10 on, PowerMux1 off, PowerMux2 on, PowerMux3 X.

010: SEM3: Sw5 on, Sw6 off, Sw11 on, PowerMux1 on, PowerMux2 X, PowerMux3 off.

011: SEM4: Sw7 on, Sw4 off, Sw12 on, PowerMux1 on, PowerMux2 X, PowerMux3 on.

100: Temperature1: Sw0 on, Sw3 on, Sw9 on, PowerMux1 off, PowerMux2 off, PowerMux3 X.

101: Temperature2: Sw1 on, Sw2 on, Sw10 on, PowerMux1 off, PowerMux2 on, PowerMux3 X.

110: Temperature3: Sw4 on, Sw7 on, Sw11 on, PowerMux1 on, PowerMux2 X, PowerMux3 off.

111: Temperature4: Sw5 on, Sw6 on, Sw12 on, PowerMux1 on, PowerMux2 X, PowerMux3 on.

D3: *n.u. was track or reset home mode select*

D4:

0: PS disabled.

1: PS enabled.

D5:

0: MotorRelay off.

1: MotorRelay on.

D7, D6: Ruler multiplexer select

00: Channel 1

01: Channel 2

10: Channel 3

11: Channel 4

SRAM selection register R/W (Address: HEX"4A"/DEC"74"):

D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	X	X	d1	d0

D1, D0: SRAM Select

00: SRAM 0, diagnostics

01: SRAM 1, logarithmic amplifier

10: SRAM 2, potentiometer position

11: SRAM 3, optical ruler position

D7-D2: NU.

FPGA Relay control register R/W (Address: HEX"42"/DEC"66"):

D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	d3	d2	d1	d0

D1, D0:

- 00: Relay12+/- both off: CH1=SEM1, CH2=SEM2.
- 01: Relay12+ on: CH1=PMT1, CH2=WireTemp2.
- 10: Relay12- on: CH1=WireTemp1, CH2= PMT1.
- 11: Relay12+ on: CH1=PMT, CH2=WireTemp2.

D3, D2:

- 00: Relay34+/- both off: CH3=SEM3, CH4=SEM4.
- 01: Relay34+ on: CH3=PMT2, CH4=WireTemp4.
- 10: Relay34- on: CH3=WireTemp3, CH4= PMT2.
- 11: Relay34+ on: CH3=PMT, CH4=WireTemp4.

D4-D7: NU.

Registers inside FPGA 2(VME Data width = 16):

FPGA FGen Clock division counter register R/W (Address: HEX"00"/DEC"0" (LW), HEX"02"/DEC"2" (HW)):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
d15	d14	d13	d12	d11	d10	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	d17	d16

D17-D0: number to divide 40 MHz clock with to obtain FGen clock.

D31-D18: NU.

FPGA Acq. Clock division counter register R/W (Address: HEX"0E"/DEC"14"):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
d15	d14	d13	d12	d11	d10	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0

D15-D0: number to divide 40 MHz clock with to obtain Acq. Clock (normally 500 kHz).

FPGA Ruler Reference Register R only (Address: HEX"04"/DEC"4" (LW), HEX"06"/DEC"6" (HW))+ reset W (Address: HEX"04"/DEC"4"):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
d15	d14	d13	d12	d11	d10	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	d17	d16

D17-D0: number of ruler pulses before occurrence of reference pulse.

D31-D18: NU.

FPGA Ruler Error Register R only (Address: HEX"08"/DEC"8" (LW), HEX"0A"/DEC"10" (HW))+ reset W (Address: HEX"08"/DEC"8"):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
d15	d14	d13	d12	d11	d10	D9	d8	d7	d6	d5	d4	d3	d2	d1	d0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	d17	d16

D17-D0: number of ruler error pulses.

D31-D18: NU.

FPGA status buffer R only (Address: HEX"12"/DEC"18"):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	X	X	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0

If d#=1:

D0: Scan

- D1: FGen busy.
- D2: FGen min.
- D3: FGen max.
- D4: End-of-stroke switch hit.
- D5: Wire reset home.
- D6: 40 MHz from BOBR signal present.
- D7: Frev from BOBR signal present.
- D8: Ext. clock signal present.
- D9: Synchronous Acquisition gate.
- D15-D10: NU.

FPGA control register R/W (Address: HEX"18"/DEC"24"):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	X	X	X	X	d7	d6	d5	d4	d3	d2	d1	d0

- D1, D0: FGen ROM mode select:
 - 00: Mode 0: linear profile for scanning slowly from switch to fast scan start position.
 - 01: Mode 1: profile for fast scan with acc, dec. and const. speed zones.
 - 10: Mode 2: linear profile for slow scan.
 - 11: Not used.
- D3, D2 clock signal source selection:
 - 00: use 40Mhz X-tal and acq. clock division counter.
 - 01: use 40 MHz from BOBR and acq. clock division counter.
 - 10: use Frev from BOBR.
 - 11: use Ext. clock.
- D4: Scan mode:
 - 0: Normal scan mode. ADC3 triggered with Frev/500kHz (<1 MHz!).
 - 1: Calibration mode. ADC3 triggered with ruler pulses.
- D5: Acq Gate:
 - 0: Acquisitions not using AcqGate.
 - 1: Acquisitions using AcqGate.
- D6: Track or reset home mode:
 - 0: Wire position track mode.
 - 1: Reset wire home mode.
- D7: FGEN address mode
 - 0: Use actual FGEN end address during a scan (HEX'FFF')
 - 1: Restrict FGEN end address to the value in the FGEN end address register *ONLY IF FGEN ROM MODE 2 IS SET*

D15-D8: NU.

FGEN end address register R/W (Address: HEX"16"/DEC"22"):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	d11	d10	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0

D11-D0: End address for the FGEN (function generator) if bit 7 in the control register is set.
 D15-D12: NU.

Registers inside FPGA 3 (VME Data width = 16):

Error register R only (Address: HEX"20"/DEC"32" (LW), HEX"22"/DEC"34" (HW)):

All HW statuses are fed into a FPGA 3 register

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
d15	d14	d13	d12	d11	d10	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0
X	X	X	X	X	X	X	d24	d23	d22	d21	d20	d19	d18	d17	d16

If all Bits 1 then OK, else:

Active error bit #	LED Display	Error message

	-	No error
D0	0	Wire broken.
D1	1	Wire hot.
D2	2	Heat sink over temperature.
D3	3	Fan current low.
D4	4	-2V VME supply missing.
D5	5	-2V internal supply missing.
D6	6	-5V internal supply missing.
D7	7	MPS voltage error NU.
D8	8	Wire1 not near home position.
D9	9	Wire2 not near home position.
D10	A	Wire3 not near home position.
D11	b	Wire4 not near home position.
D12	C	Potentiometer1 voltage too low.
D13	d	Potentiometer2 voltage too low.
D14	E	Potentiometer3 voltage too low.
D15	F	Potentiometer4 voltage too low.
D16	I	Ruler status bad.
D17	J	PG1 - Internal power for FPGA1 bad
D18	L	PG11 - PLL1 power for FPGA1 bad
D19	M	PG12 - PLL2 power for FPGA1 bad
D20	n	PG2 - Internal power for FPGA2 bad
D21	o	PG21 - PLL1 power for FPGA2 bad
D22	P	PG22 - PLL2 power for FPGA2 bad
D23	r	PG31 - PLL2 power for FPGA3 bad
D24	t	PG32 - PLL2 power for FPGA3 bad

Conditions indicated by individual LED's controlled by FPGA:

- 1: Wire1 not near home position while not scanning.
- 2: Wire2 not near home position while not scanning.
- 3: Wire3 not near home position while not scanning.
- 4: Wire4 not near home position while not scanning

If there are no errors, then '-' is displayed. If 1 error, then a single fixed error number is displayed. If several errors occur, then the error characters are stepped through with a 0.5 sec interval. One 7-segment LED display is enough to cope with the error reporting. The decoding is done inside the FPGA and the LED driver does not contain any decoder.

As the number of the errors exceeds 15, letters are used as well, in conjunction with <http://www.twyman.org.uk/Fonts/> :

```

AbCdEFgHI jKlñ
nOPq-rStUv'WYz
0 1234567890

```

Some characters cannot be used because they are duplicated (g (9), I (1), k (h), o (0), s (5), x (h)). However, some of them (g, h, o, s) may be modified keeping a reasonable readability.

Registers outside FPGA:

ADC0 (AD7938) control register *W only* (Address: HEX"90"/DEC"144"):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	PM1	PM0	COD	REF	A2	A1	A0	MD1	MD0	SH	SQ	RG

D0: RANGE 0: Input range from 0 to V_{ref} ; 1: Input range from 0 to $2xV_{ref}$.

D1, D2: SEQ, SDWN

00: sequencing not used.

01: shadow register used.

10: sequence function not interrupted after write operation.

11: sequencing is used.

D4, D3: MODE1, MODE0

00: 8 single-ended channels.

01: 4 fully differential channels.

10: 4 pseudo differential channels.

11: 7 pseudo differential channels.

D7, D6, D5: ADD2, ADD1, ADD0 : channel addresses.

D8: REF 0: external reference used; 1: internal reference used.

D9: CODING 0: output is straight binary; 1: output is twos complement.

D10, D11: PM0, PM1

00: Normal power mode – all circuitry is powered.

01: auto shutdown – enters in full shutdown after each conversion.

10: Auto standby – all circuitry is powered down except for the reference and buffer.

11: Full shutdown – all circuitry is powered down.

D15-D12: NU.

ADC0 (AD7938) shadow register *W only* (Address: HEX"90"/DEC"144"):

Beware that this register can only be filled on the following write operation after the SH-bit in the ADC0 control register has been set.

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	X	X	X	X	V_{in7}	V_{in6}	V_{in5}	V_{in4}	V_{in3}	V_{in2}	V_{in1}	V_{in0}

D7-D0: Enable the channels to be converted during a sequence.

D15-D8: NU.

ADC1 (AD7484) offset register *W only* (Address: HEX"92"/DEC"146"):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	0	0	0	d11	d10	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0

D11-D0: Value added to conversion result. To be filled as twos complement.

D14-D12: Set to 0.

D15: NU.

The default contents of the offset register are 0. If the offset register contains any value other than 0, the contents of the register are added to the SAR result at the end of conversion. To write to the offset register, a 15-bit word is written to the AD7484 with the 12 LSBs containing the offset value in twos complement format. The 3 MSBs must be set to 0. The offset value must be within the range -1310 to $+1310$, corresponding to an offset from -200 mV to $+200$ mV.

DAC00 (AD5582) Logarithmic amplifier test current *R/W* (Address: HEX"80"/DEC"128"):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	d11	d10	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0

D11-D0: Binary value representing an output voltage ranging from -2.5 V to $+2.5$ V.

D15-D12: NU.

A test current for the logarithmic amplifier is set by the output voltage of this DAC output. This has to be a negative voltage, due to the polarization of a diode in series. When the logarithmic amplifier is to be used in normal operation, this output value should be set to 0 V (HEX"800").

DAC01 (AD5582) Wire current R/W (Address: HEX"82"/DEC"130"):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	d11	d10	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0

D11-D0: Binary value representing an output voltage ranging from -2.5 V to +2.5 V.
D15-D12: NU.

A test current for the wire temperature measurements is set by the output voltage of this DAC output. This current is necessary during the temperature measurements, and the wire current can be deduced from the following formulae:

$$I_{out} = -\frac{R_{98} \cdot V_{DAC01}}{R_{79} \cdot R_{106}} = -\frac{2M\Omega \cdot V_{DAC01}}{1M\Omega \cdot 2k\Omega} = -\frac{V_{DAC01}}{1k\Omega}$$

DAC02/03 (AD5582) Scan Gate Lower/Upper limit R/W (Address: HEX"84"/DEC"132"):

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	d11	d10	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0

D11-D0: Binary value representing an output voltage ranging from -2.5 V to +2.5 V.
D15-D12: NU.

DAC02/03 values represent the voltages for the respectively lower/upper voltage limits of the scan gate. When the scan gate bit of the FPGA2 Control Register is set, acquisitions are only performed when the potentiometer voltage is within the lower and upper limit of the scan gate.

D. VME memory mapping

VME-16 Address Mapping (AM: x29 or x2D)

	Address		Write Data width	Write Function	Read Data width	Read Function
	Hex	Dec				
FPGA2	00	0	16bits	FGEN Clock division counter register Lo	16 bits	FGEN Clock division counter register Lo
FPGA2	02	2	2 bits	FGEN Clock division counter register Hi	2 bits	FGEN Clock division counter register Hi
FPGA2	04	4	0	Clear quadrature decoder reference register	16 bits	Ruler Reference Register Lo
FPGA2	06	6	-	N.U.	2 bits	Ruler Reference Register Hi
FPGA2	08	8	0	Clear quadrature decoder error counter	16 bits	Ruler error register Lo
FPGA2	0A	10	-	N.U.	2 bits	Ruler error register Hi
FPGA2	0C	12	0	Clear quadrature decoder U/D counter	-	N.U.
FPGA2	0E	14	16 bits	Acq. Clock division counter register	16 bits	Acq. Clock division counter register
FPGA2	10	16	0	Clear Memory address counter	-	N.U.
FPGA2	12	18	0	Set Fgen ROM address counter to FFF	10 bits	Status buffer R only
FPGA2	14	20	0	Clear Fgen ROM address counter	12 bits	FG ROM 4k x 12 bits (*)
FPGA2	16	22	12 bits	FGEN end address value	12 bits	FGEN end address value
FPGA2	18	24	8 bits	Control register	8 bits	Control register
FPGA2	1A	26	0	Start motion	-	N.U.
FPGA2	1C	28	0	Motion reset	-	N.U.
FPGA2	1E	30	0	FPGA II reset (also set by master reset)	-	N.U.
FPGA3	20	32	-	N.U.	16 bits	Read error register Lo
FPGA3	22	34	-	N.U.	8 bits	Read error register Hi
FPGA3	3E	62	0	FPGA III reset (also set by master reset)	-	N.U.
FPGA1	40	64	0	Clear Acquisition Address Counter	-	N.U.
FPGA1	42	66	4 bits	Relay control register	4 bits	Relay control register
FPGA1	44	68	8 bits	I/O register Lo	8 bits	I/O register Lo
FPGA1	46	70	8 bits	I/O register Hi	8 bits	I/O register Hi
FPGA1	48	72	8 bits	Switch control register	8 bits	Switch control register
FPGA1	4A	74	2 bits	SRAM selection register	2 bits	SRAM selection register
FPGA1	4C	76	-	N.U.	8 bits	Versions register
DAC00	80	128	12 bits	DAC00: Log amp test current	12 bits	DAC00: Log amp test current
DAC01	82	130	12 bits	DAC01: Wire current	12 bits	DAC01: Wire current
DAC02	84	132	12 bits	DAC02: Scan Gate Lower limit	12 bits	DAC02: Scan Gate Lower limit
DAC03	86	134	12 bits	DAC03: Scan Gate Upper limit	12 bits	DAC03: Scan Gate Upper limit
ADC0	90	144	12/8 bits	ADC0 control/shadow register(ENABLE)**	12 bits	ADC0 Misc data
ADC1	92	146	14+1(***) bits	ADC1 Log.amp offset register (ENABLE)	14+1(***) bits	ADC1: Log. Amp acq.
ADC2	94	148	-	N.U.	16 bits	ADC2: Potentiometer acq.
All FPGAs	FE	254	0	Master reset		

Reserve addresses 64-127 (HEX"40-7F") for FPGA I functions
 Reserve addresses 0-31 (HEX"00-1F") for FPGA II functions
 Reserve addresses 32-63 (HEX"20-3F") for FPGA III functions

(*): address counter increments at every read ("FIFO mode")
 (**): If the shadow bit is set in control register, then the next write operation is the shadow data.
 (***) MSB is an overrange bit, indicates if in signal is out of range

If data pack is bigger than datawidth of the VMEbus:
 Low words addressed where VME_ADDRESS1 = 0,
 high words addressed where VME_ADDRESS1 = 1

} VME_ADDRESS1 = UWORD

VME-24 Address Mapping

AM : x39 or x3D, AM (BLT): x3B or x3F

SRAM Register	SRAM Start Address	SRAM Data width	Function (R only, values written during a scan)
00	0	256k x 16 bits	SRAM1: Log. Amp data
01	0	256k x 16 bits	SRAM2: Misc. acq. data
10	0	256k x 16 bits	SRAM3: Potentiometer data
11	0	256k x 16 bits	SRAM4: Quadrature decoder data

E. Matlab function generation script

```
%*****  
%****  
%**** Creates wire scanner motion control function vectors and ****  
%**** generates the appropriate memory initialization files. ****  
%****  
%**** USAGE: ****  
%**** Set the Data Width (DW) and the Address Width (AW) only, then ****  
%**** run the script in matlab to generate the .mif files. ****  
%****  
%*****  
  
clf;  
clc;  
clear;  
  
DW = 12;  
AW = 12;  
  
Ymax = 2^DW - 1;  
Xmax = 2^AW - 1;  
Yoffset = round(0.05 * Ymax);  
Yrange = Ymax - 2*Yoffset;  
Ymid = Yrange / 2;  
Xmid = Xmax / 2;  
Yone = Ymid - ((0.25*Ymax)/2);  
Xone = (2*Xmid*Yone)/(Ymid+Yone);  
Xtwo = Xmax - Xone;  
  
vtX = [0:1:Xmax];  
  
%*****  
%****  
%**** Offsetting function ****  
%****  
%*****  
  
funcOffset = (Yoffset/(Xmax+1))*vtX;  
plot(vtX, (130/(Ymax+1))*funcOffset);  
xlabel('Time (clock cycles)');  
ylabel('Position [mm]');
```

```

fid = fopen('startfunction.mif','w');
fprintf(fid,'WIDTH=%d;\nDEPTH=%d;\n\nADDRESS_RADIX=UNS;\nDATA_RADIX=UNS;\n\n
CONTENT BEGIN\n', AW, Xmax+1);
for m = 0:Xmax
    fprintf(fid,'%d : %d; \n', m, round(funcOffset(m+1)));
end fprintf(fid,'END;\n'); fclose(fid);

%*****
%****
%**** Main function with offset
%****
%*****

a = Yone / Xone^2; b = 2*a*Xone; c = Yone - (b*Xone);

startQuad = a*(vtX.^2) + Yoffset;
midLinear = (b*vtX) + c + Yoffset;
endQuad = -a*((vtX-Xmax).^2) + Yrange + Yoffset;

Y(1:round(Xone)) = startQuad(1:round(Xone));
Y(round(Xone)+1:round(Xtwo)) = midLinear(round(Xone)+1:round(Xtwo));
Y(round(Xtwo)+1:Xmax+1) = endQuad(round(Xtwo)+1:Xmax+1);

Y_speed = diff(Y); Y_acceleration = diff(Y_speed);

%DIGITAL
quantized = round(Y);

figure subplot(2,2,[1 3]);
plot(vtX, (130/(Ymax+1))*Y, vtX, (130/(Ymax+1))*quantized, '.');
    xlabel('Time [clock cycles]'); ylabel('Position [mm]')
%    title('Position')

subplot(2,2,2); plot(vtX(1:Xmax), (130/(Ymax+1))*Y_speed);
    xlabel('Time (clock cycles)'); ylabel('Speed [mm/cc]')
%    title('Speed')

subplot(2,2,4); plot(vtX(1:Xmax-1), (130/(Ymax+1))*Y_acceleration);
    xlabel('Time (clock cycles)'); ylabel('Acceleration [mm/cc^2]')
%    title('Acceleration')

fid = fopen('function.mif','w');
fprintf(fid,'WIDTH=%d;\nDEPTH=%d;\n\nADDRESS_RADIX=UNS;\nDATA_RADIX=UNS;\n\n
CONTENT BEGIN\n', AW, Xmax+1);
for m = 1:Xmax+1
    fprintf(fid,'%d : %d; \n', m-1, quantized(m));
end fprintf(fid,'END;\n'); fclose(fid);

%*****

```

```

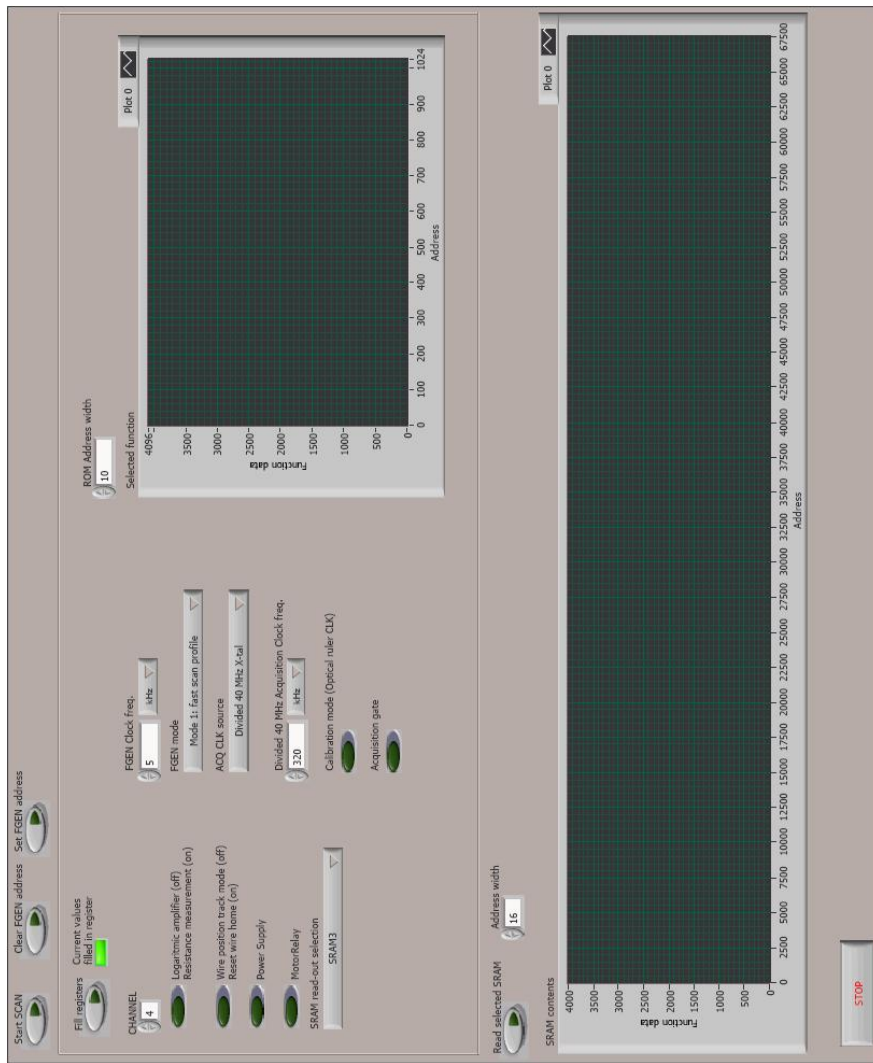
%****                                     ****
%**** Linear function                     ****
%****                                     ****
%*****
funcOffset = ((Ymax+1)/(Xmax+1))*vtX;
figure plot(vtX,(130/(Ymax+1))*funcOffset); xlabel('Time (clock cycles)');
ylabel('Position [mm]');

fid = fopen('linearfunction.mif','w');
fprintf(fid,'WIDTH=%d;\nDEPTH=%d;\n\nADDRESS_RADIX=UNS;\nDATA_RADIX=UNS;\n\n
CONTENT BEGIN\n', AW, Xmax+1);
for m = 0:Xmax
    fprintf(fid,'%d : %d; \n', m, round(funcOffset(m+1)));
end fprintf(fid,'END;\n'); fclose(fid);

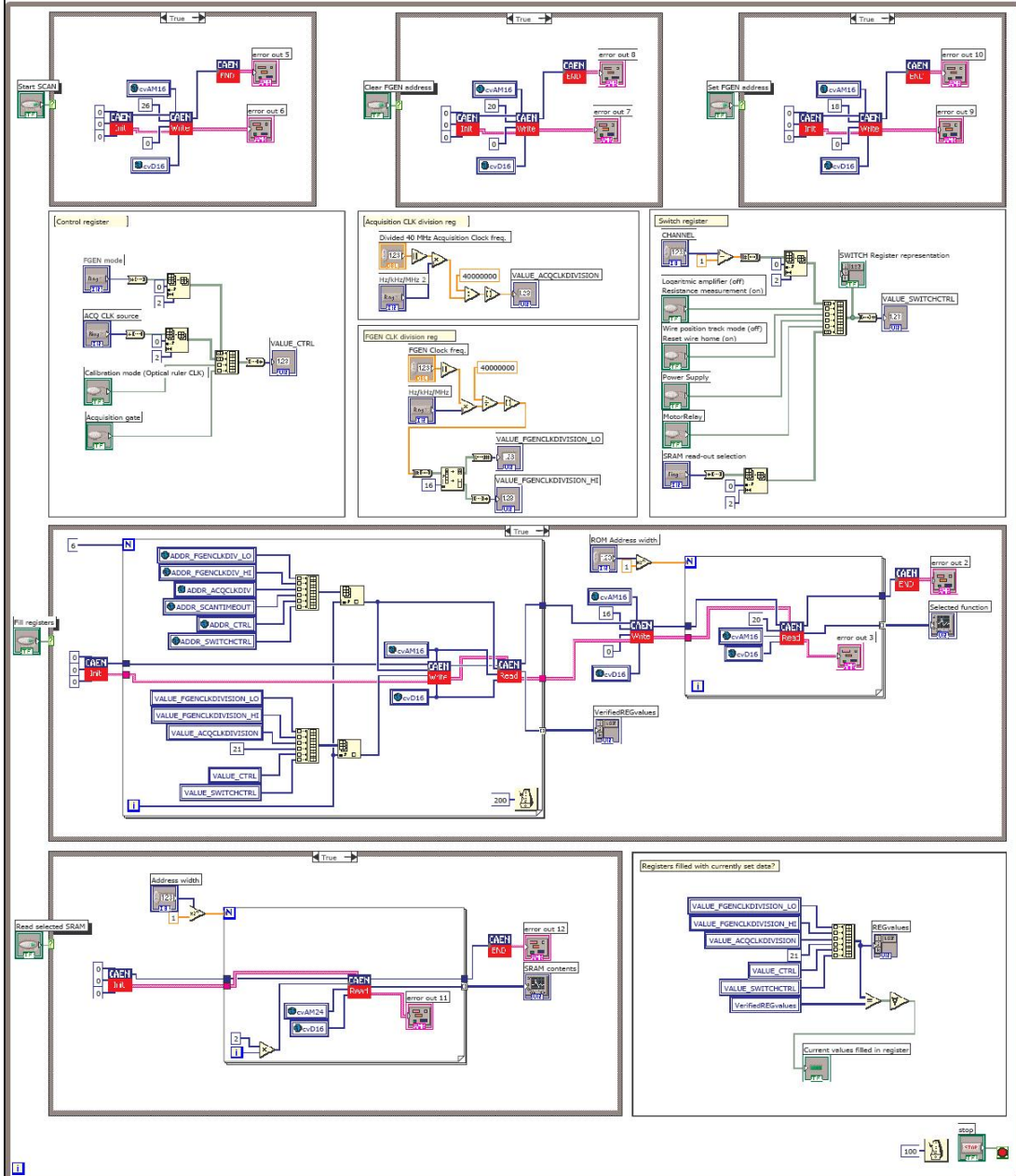
```

F. LabVIEW VME access program

F.1. Front panel, user interface



F.2. Block diagram



G. LabVIEW testbench for WSMCC

G.1. FPGA registers check

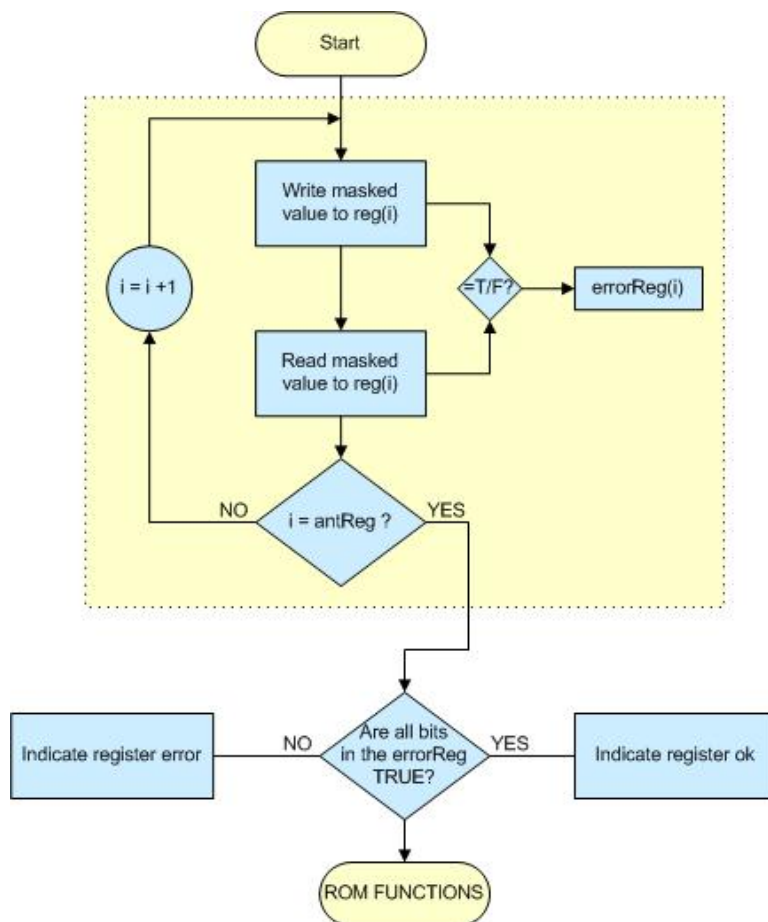


Figure G.1.: A flowchart which shows the idea of how to check the FPGA registers using a loop and an address array.

G.1. FPGA REGISTERS CHECK

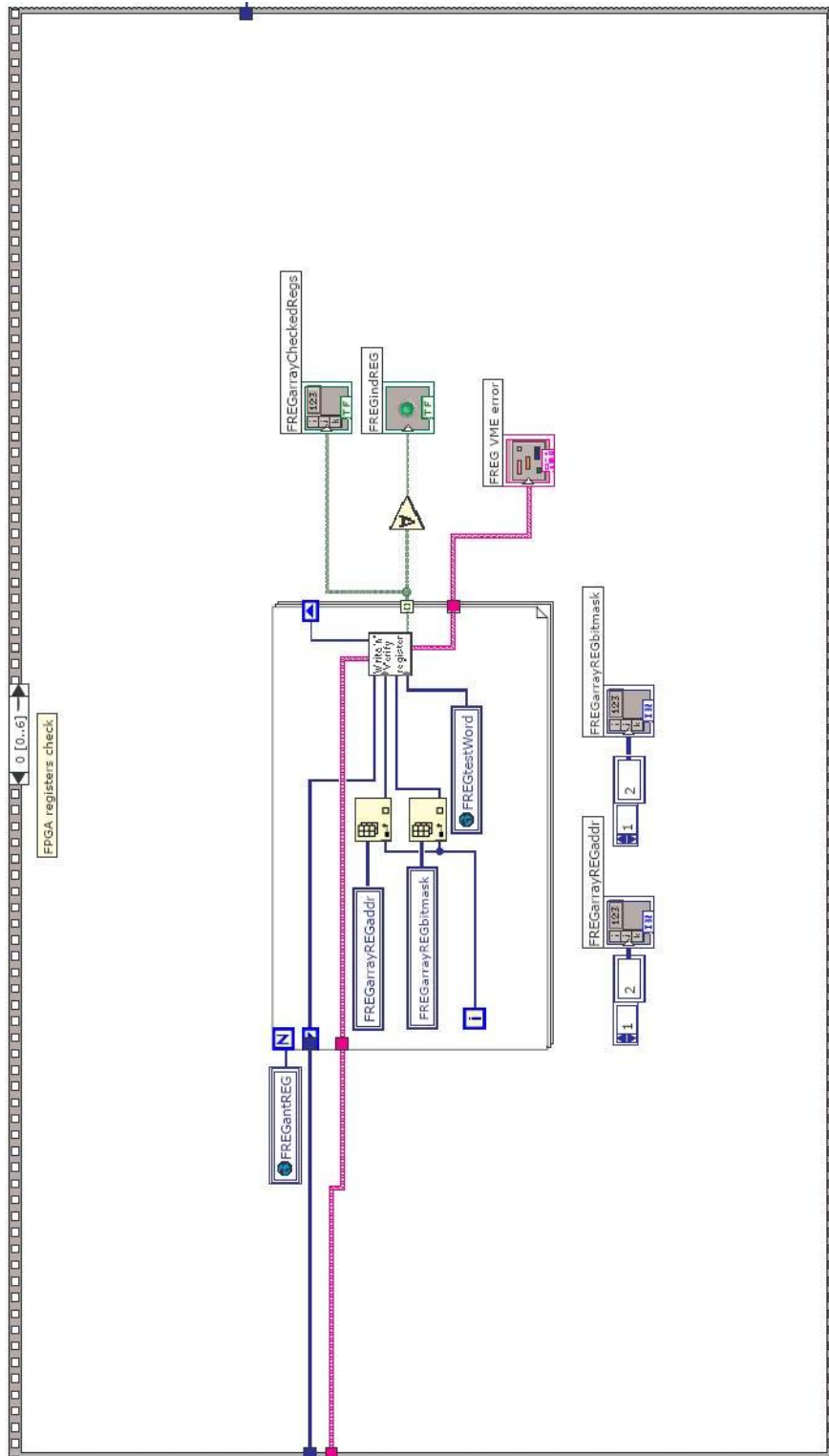


Figure G.2.: From flowchart to LabVIEW block diagram.

G.2. ROM functions check

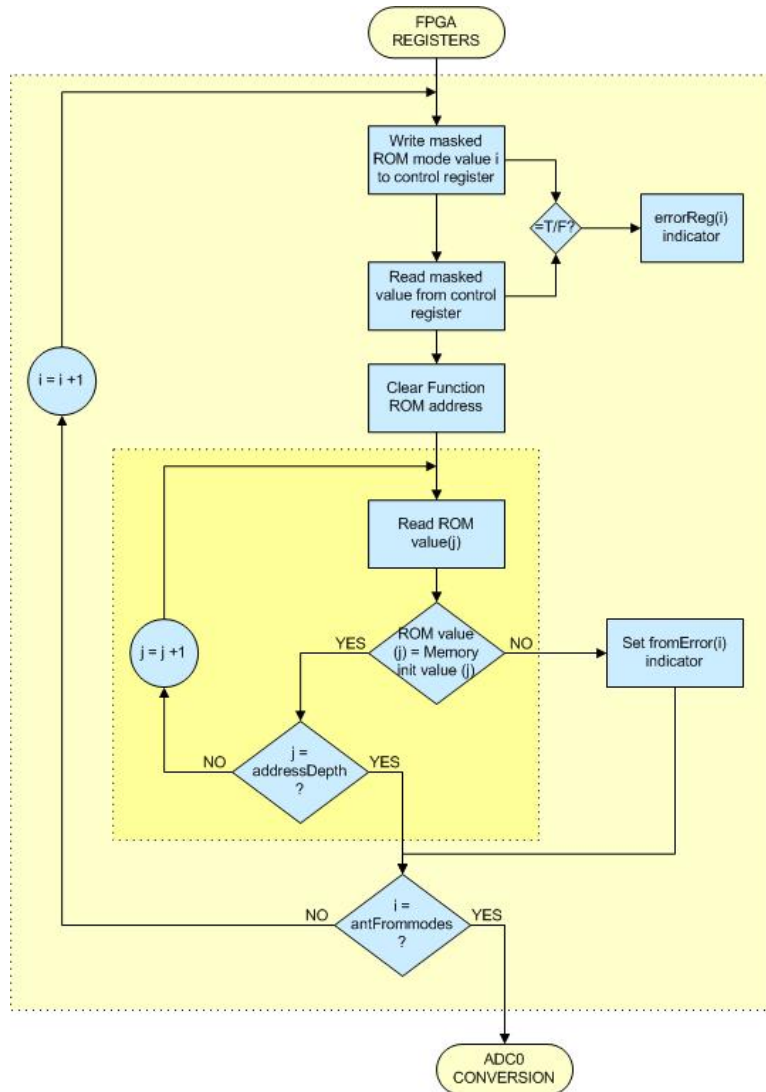


Figure G.3.: A flowchart which shows the idea of how to use loops and initial datasheet values to confirm that the functions stored in the FPGA ROM is correct.

G.2. ROM FUNCTIONS CHECK

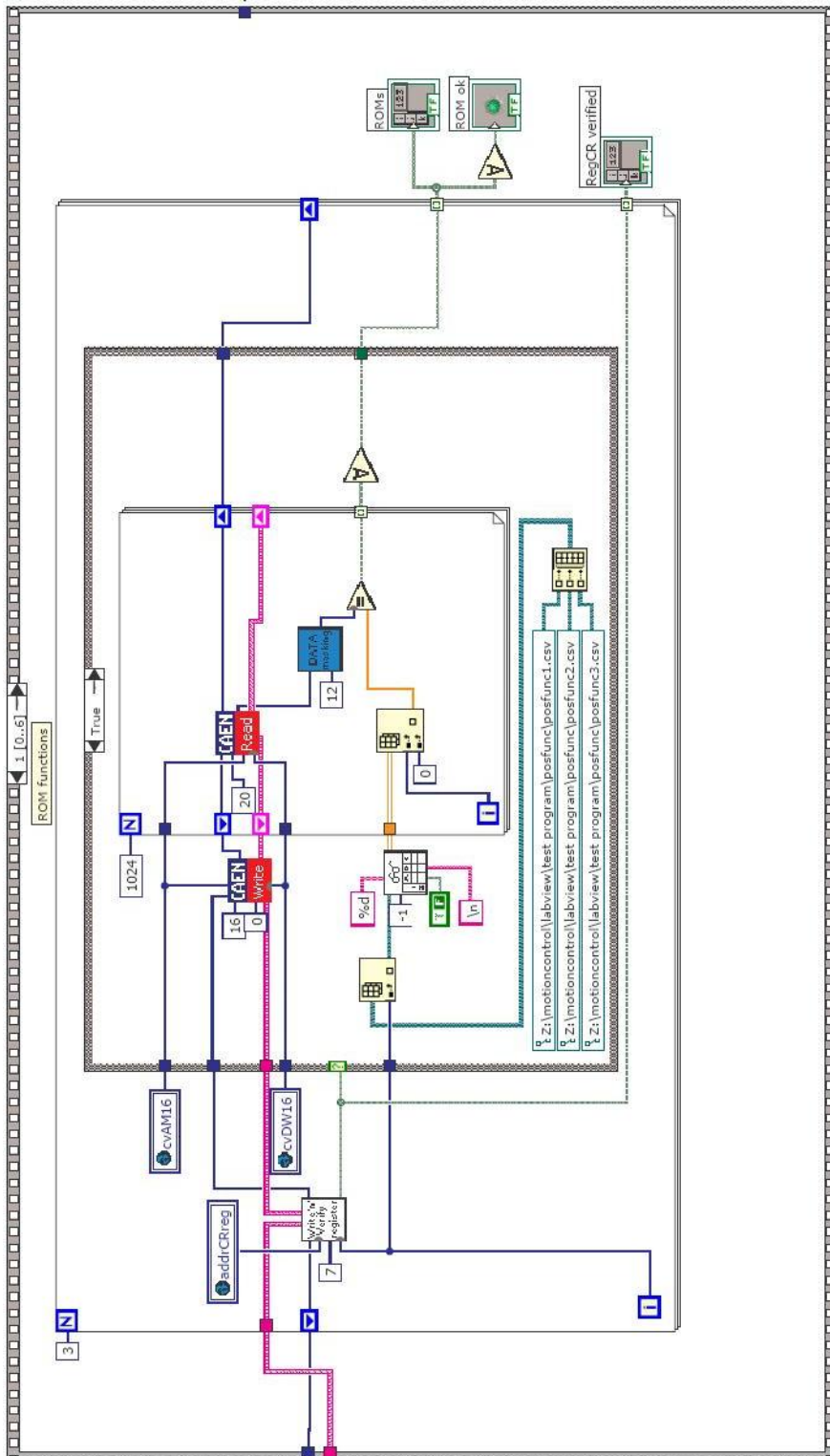


Figure G.4.: From flowchart to LabVIEW block diagram.

G.3. ADC0 conversion check

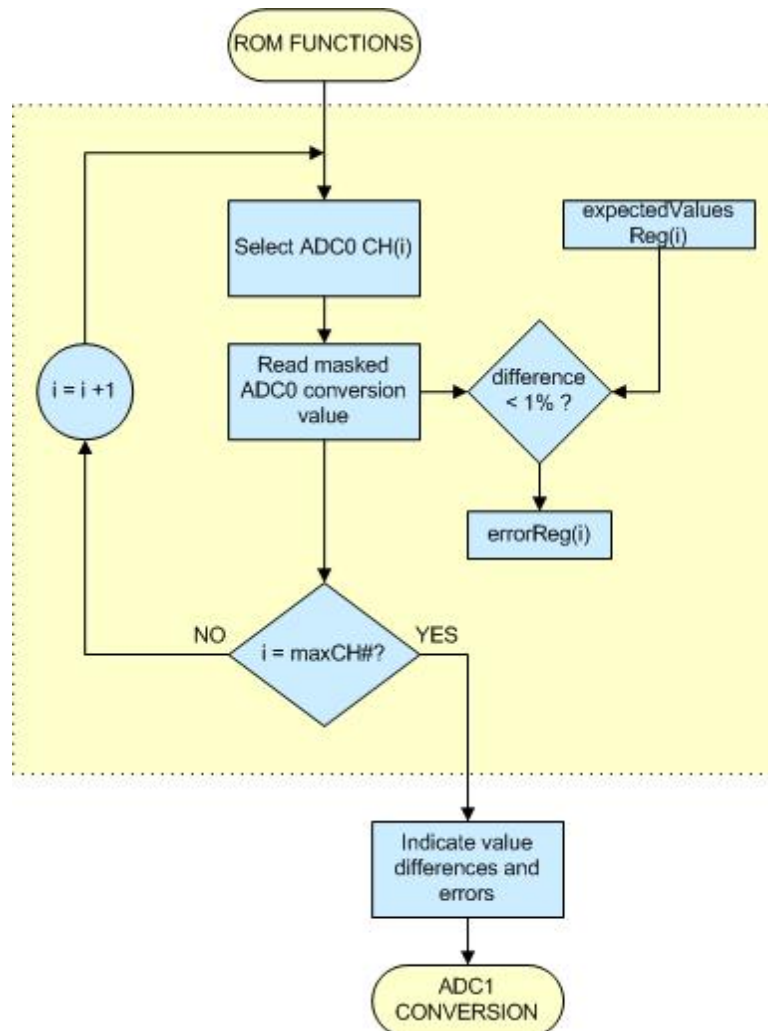


Figure G.5.: Flowchart shows the idea for how to check the ADC0 conversions, using a loop to check several channels sequentially.

G.3. ADC0 CONVERSION CHECK

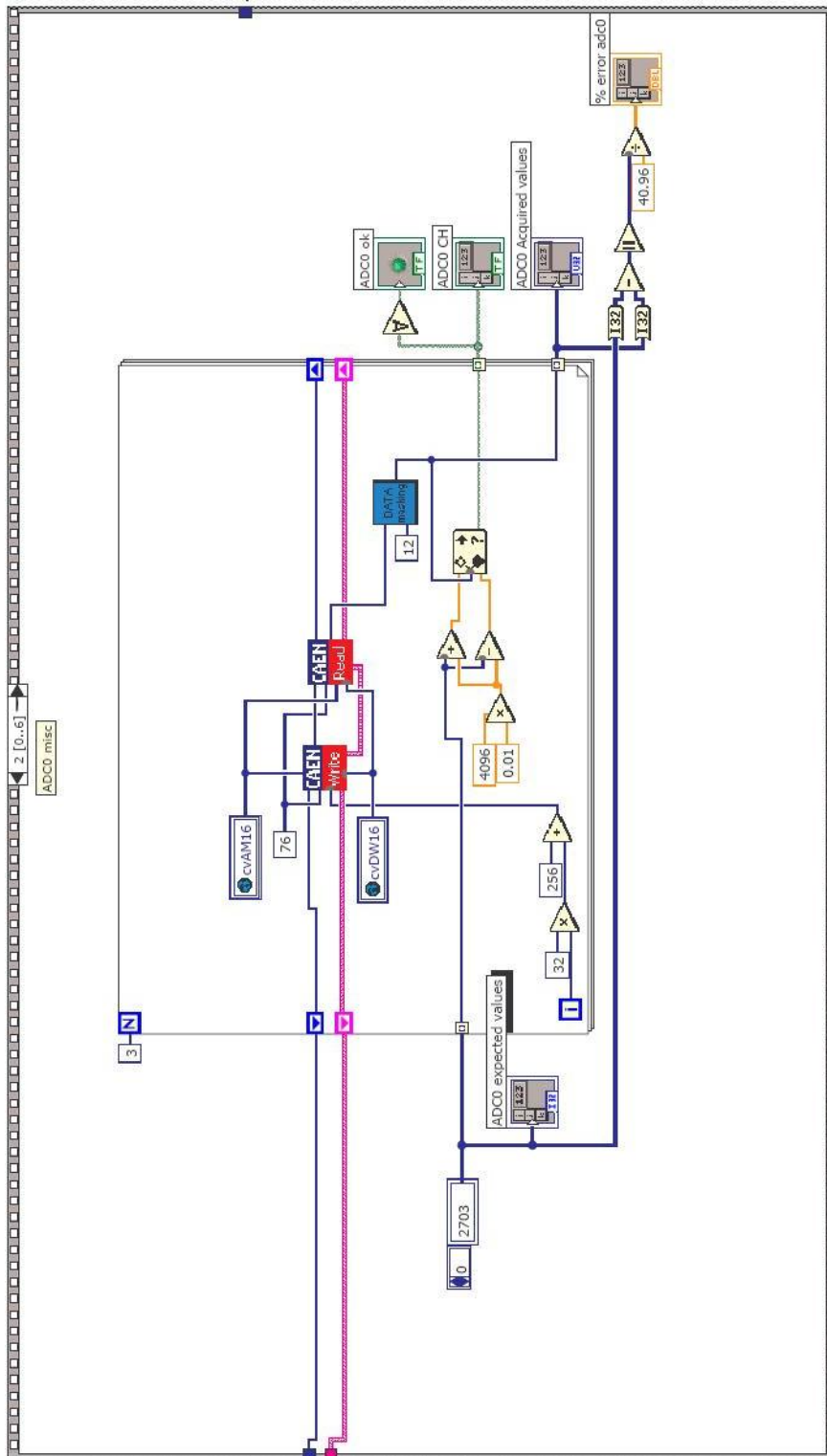


Figure G.6.: From flowchart to LabVIEW block diagram..

G.4. ADC1 conversion check

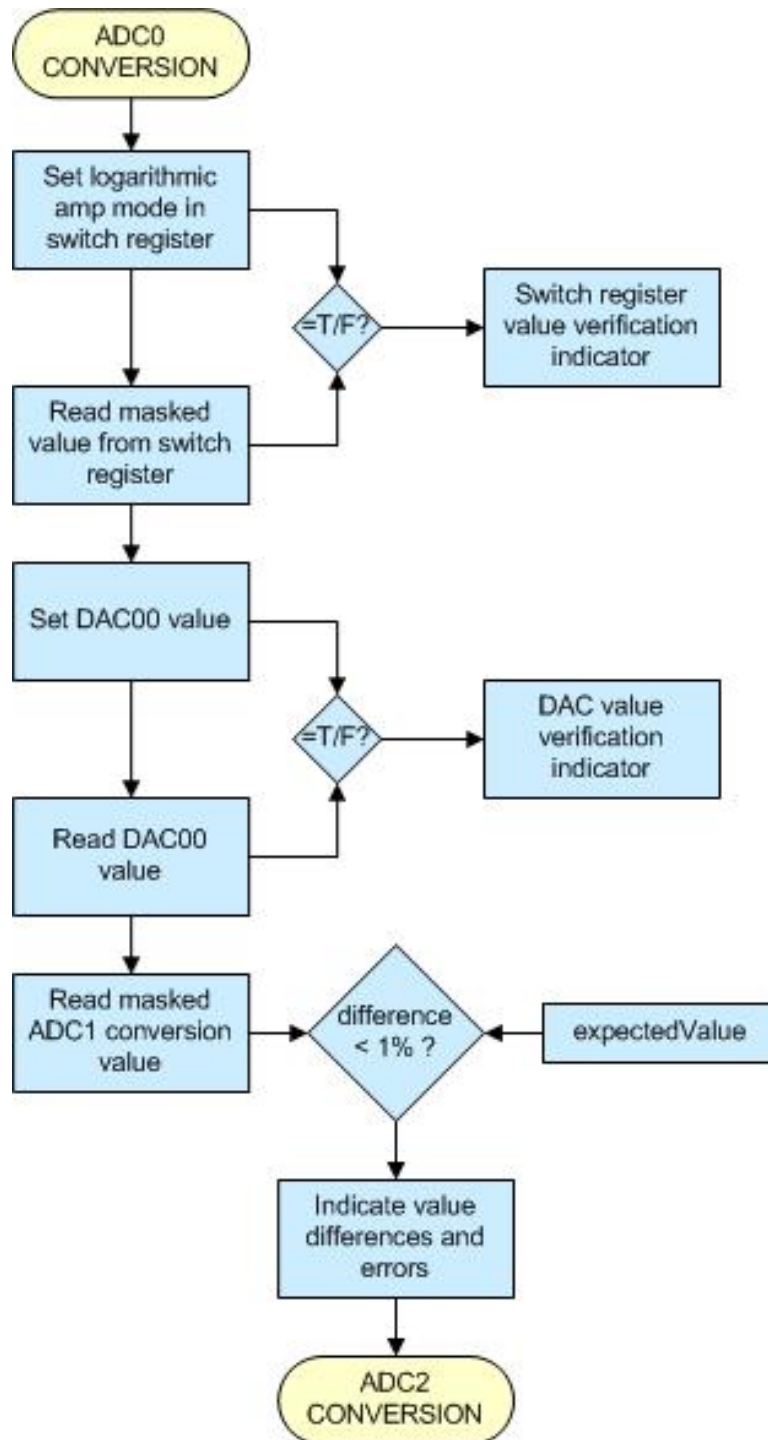


Figure G.7.: Flowchart shows the idea for how to check the ADC1 conversion.

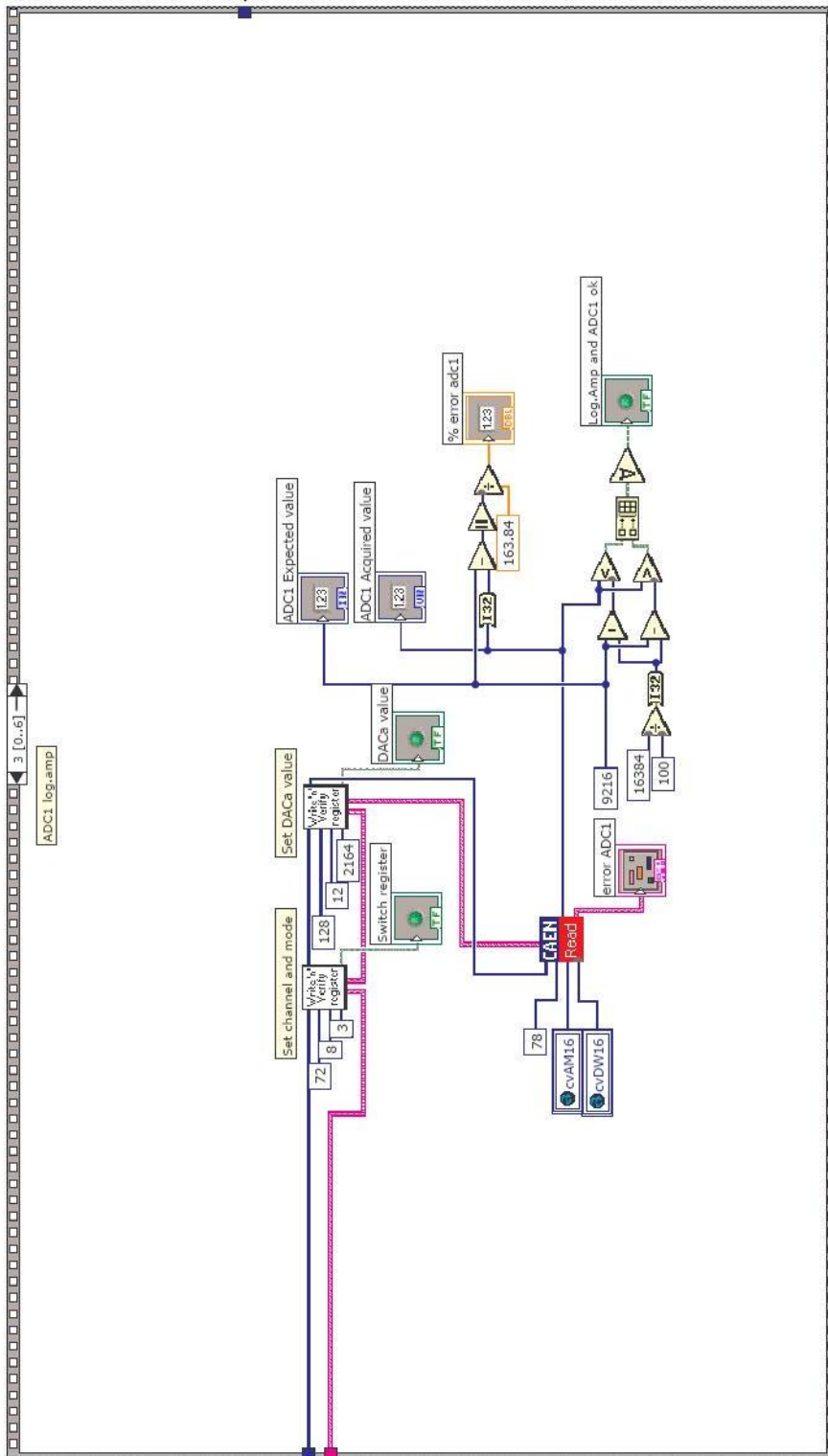


Figure G.8.: From flowchart to LabVIEW block diagram..

G.5. ADC2 conversion check

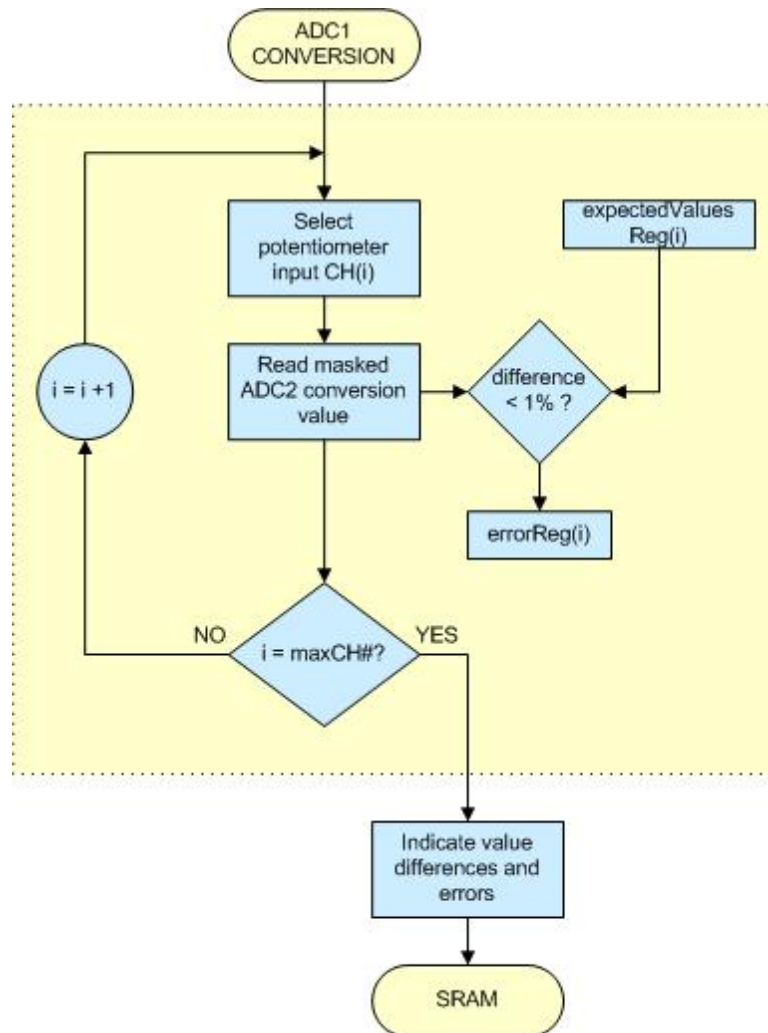


Figure G.9.: Flowchart shows the idea for how to check the ADC2 conversion.

G.5. ADC2 CONVERSION CHECK

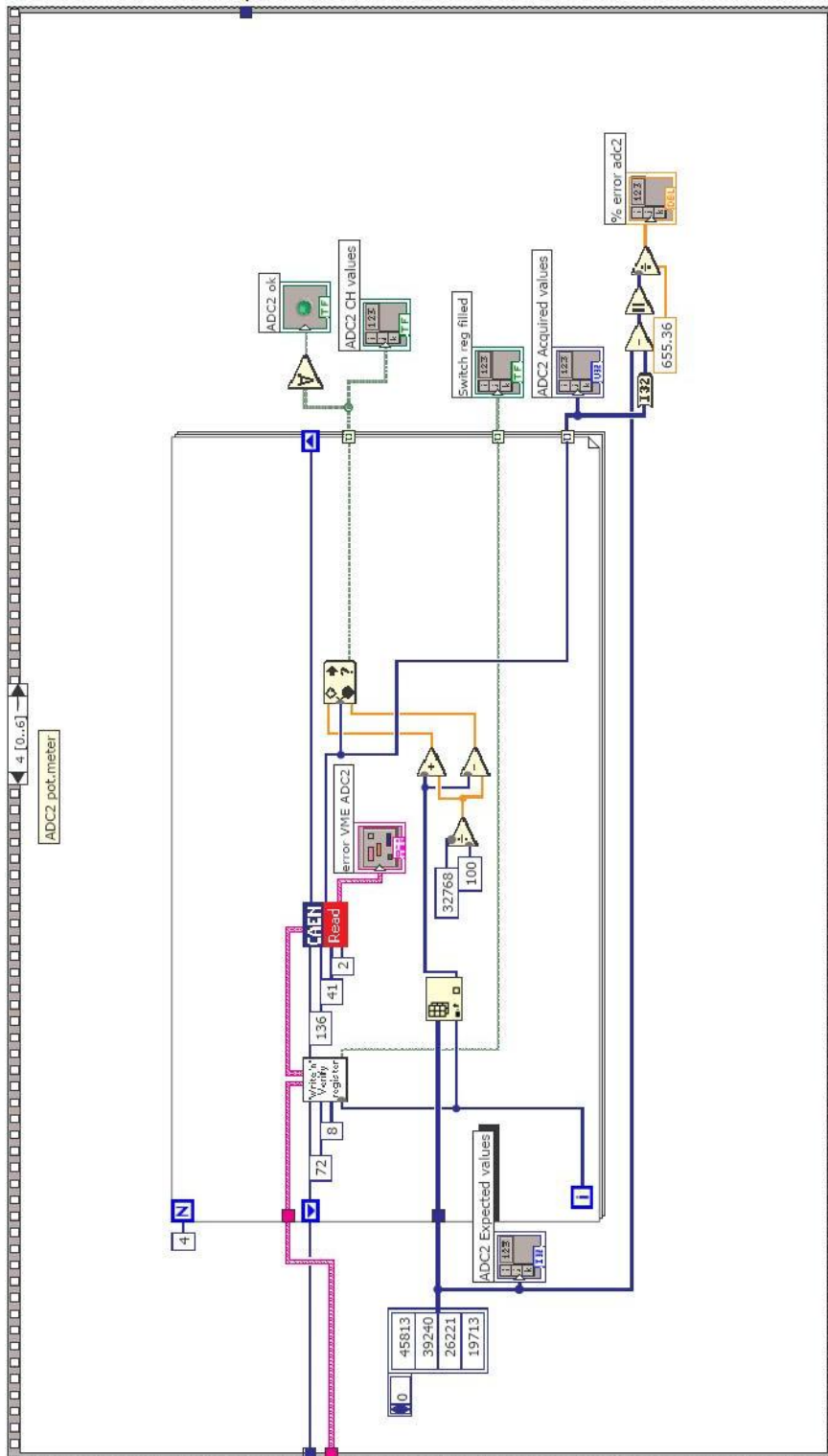


Figure G.10.: From flowchart to LabVIEW block diagram..

G.6. SRAM check

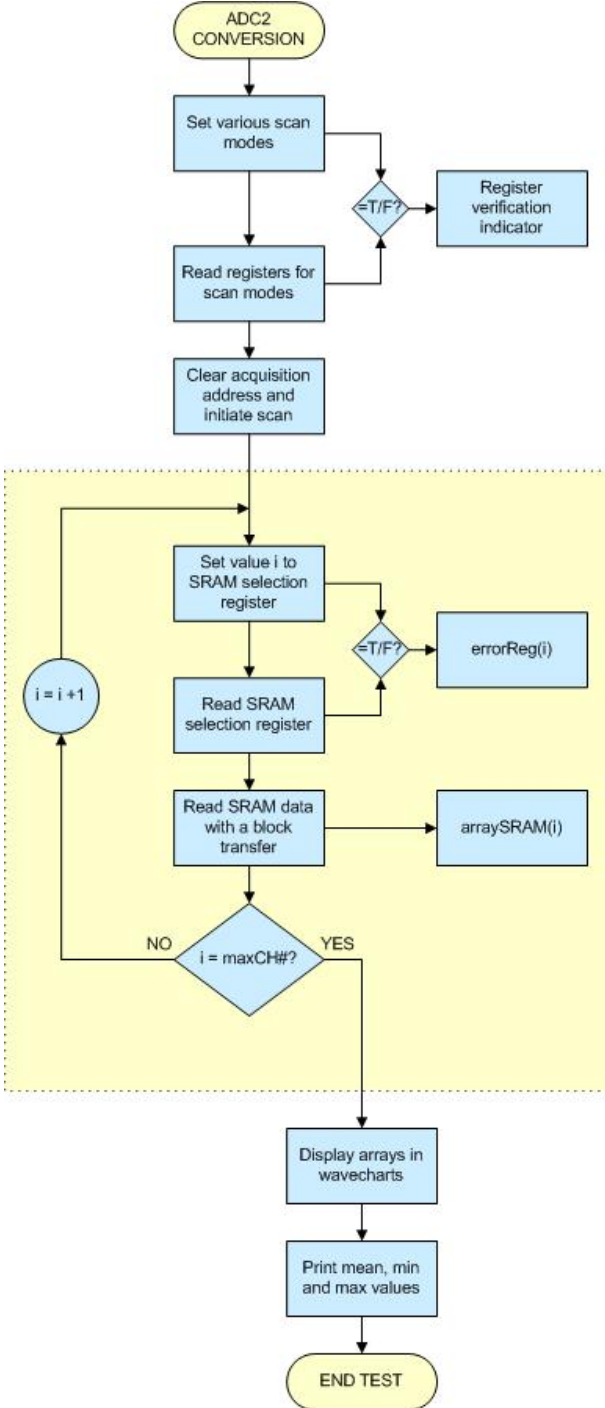


Figure G.11.: This flowchart shows the idea for how perform a scan and read out the acquired values for diagnosis and graphical display.

G.6. SRAM CHECK

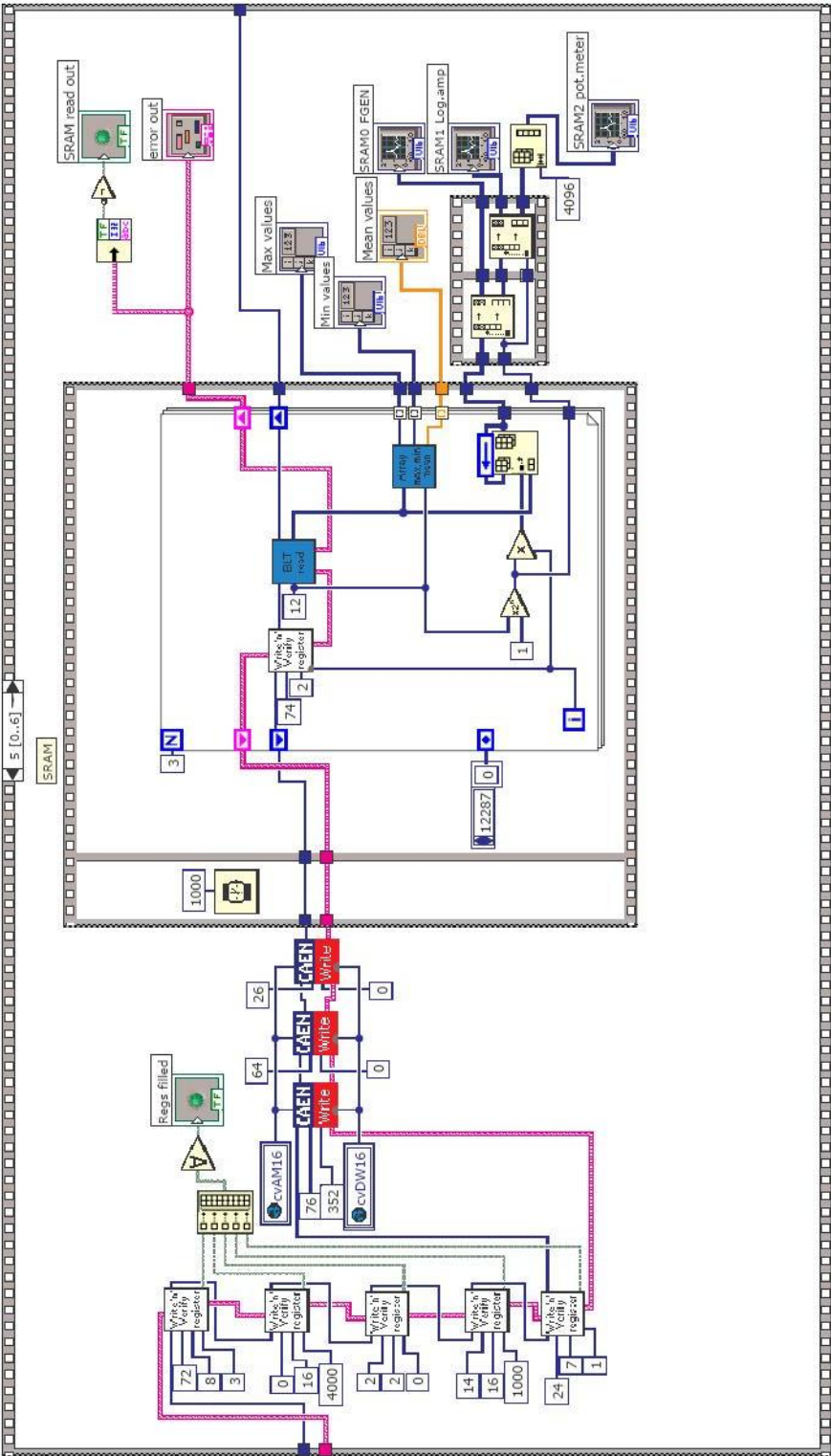
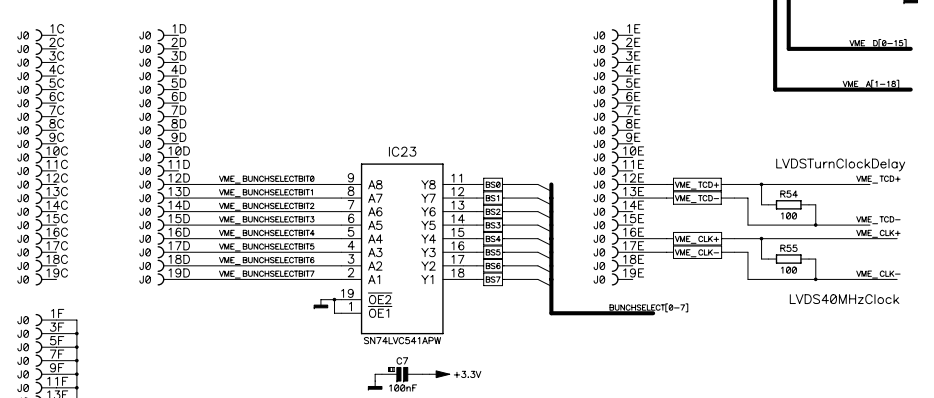
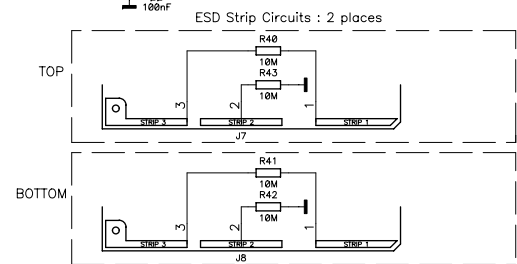
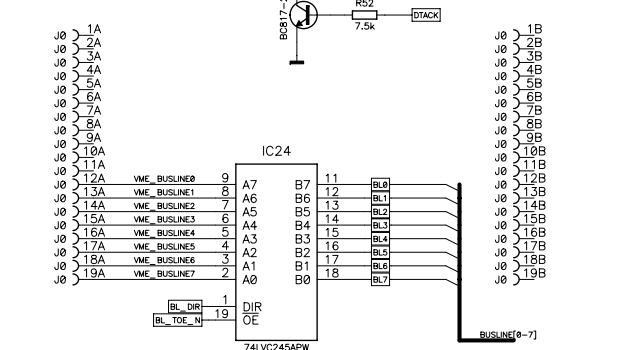
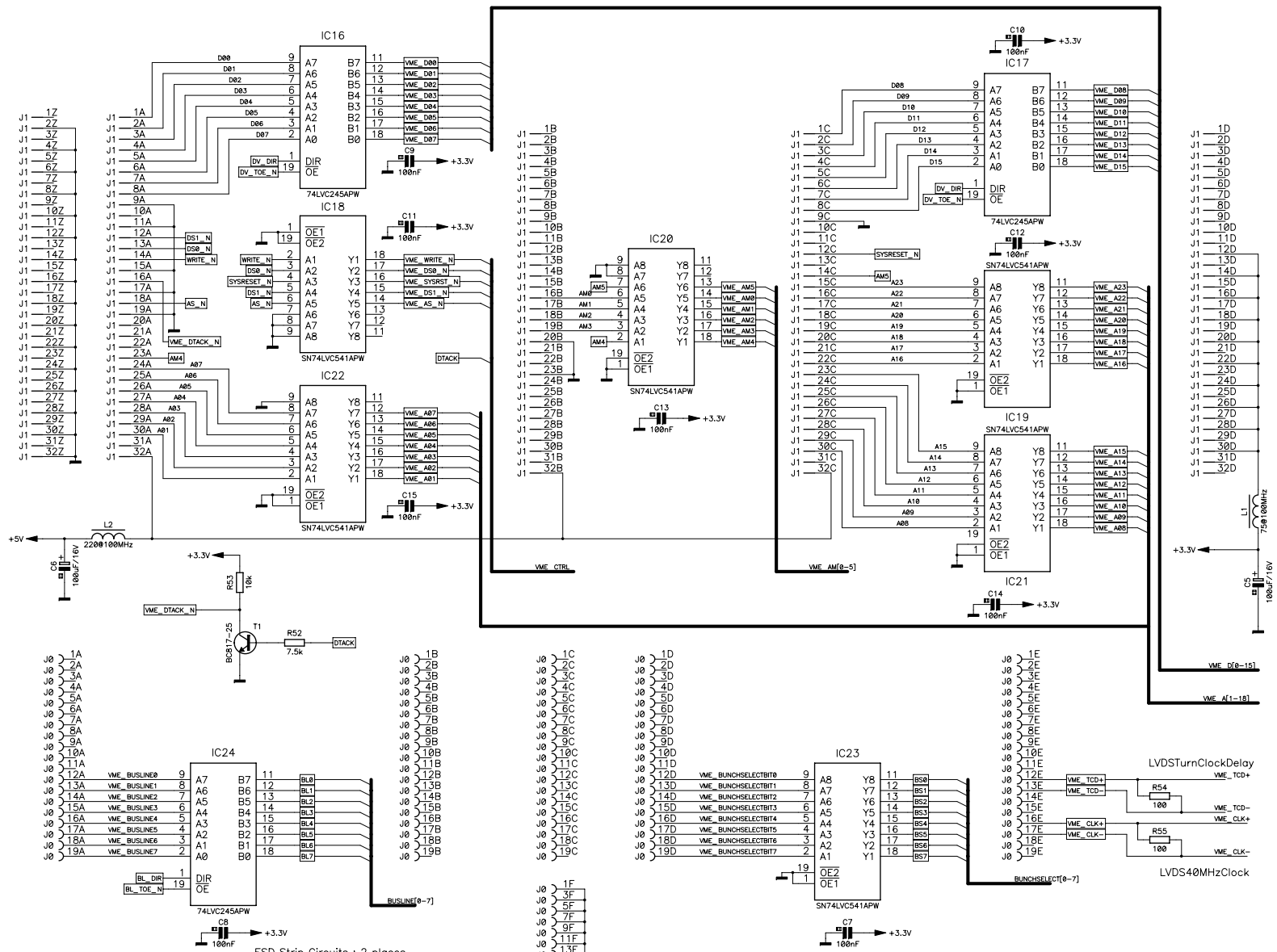


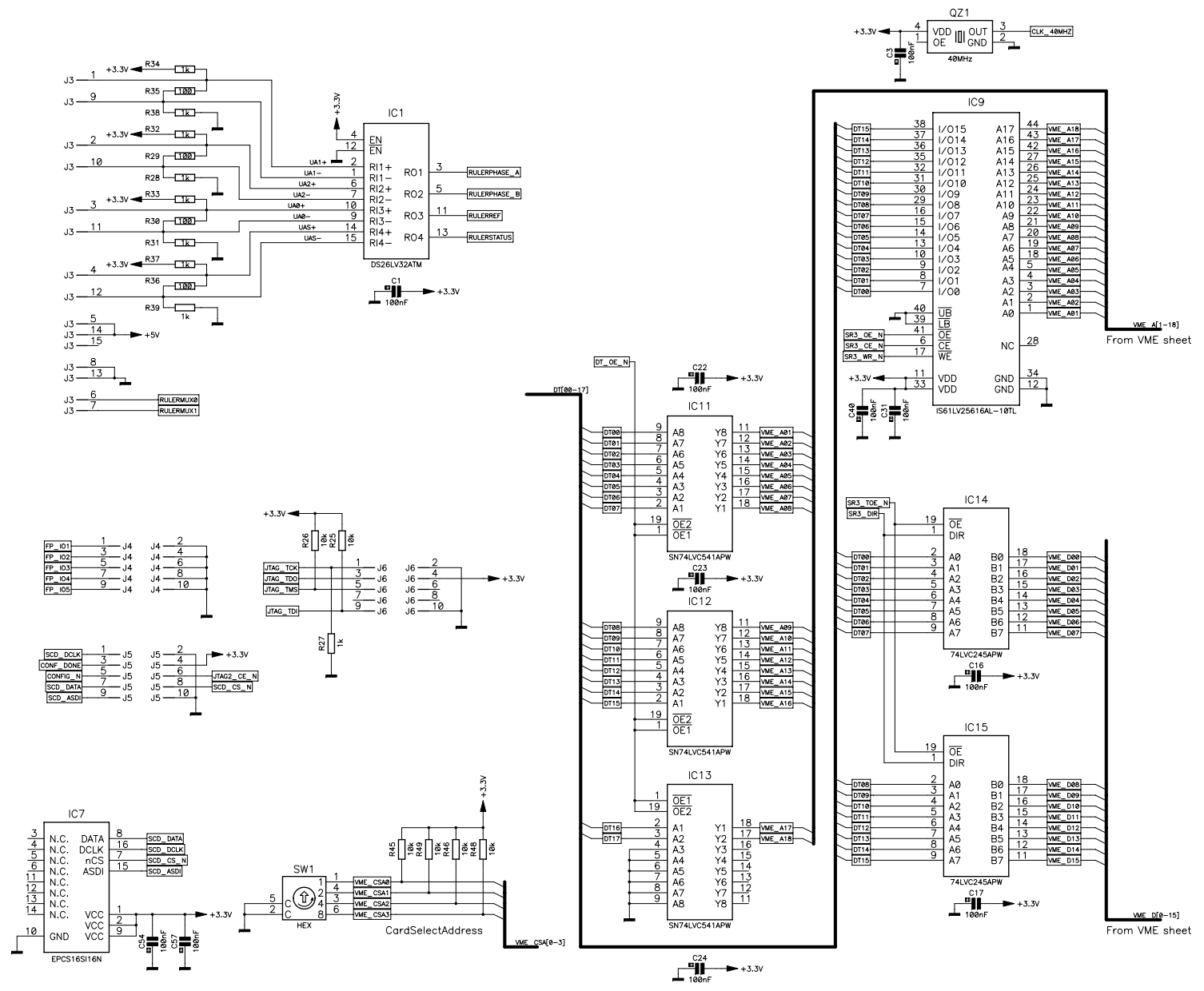
Figure G.12.: From flowchart to LabVIEW block diagram..

H. FPGA Prototype card

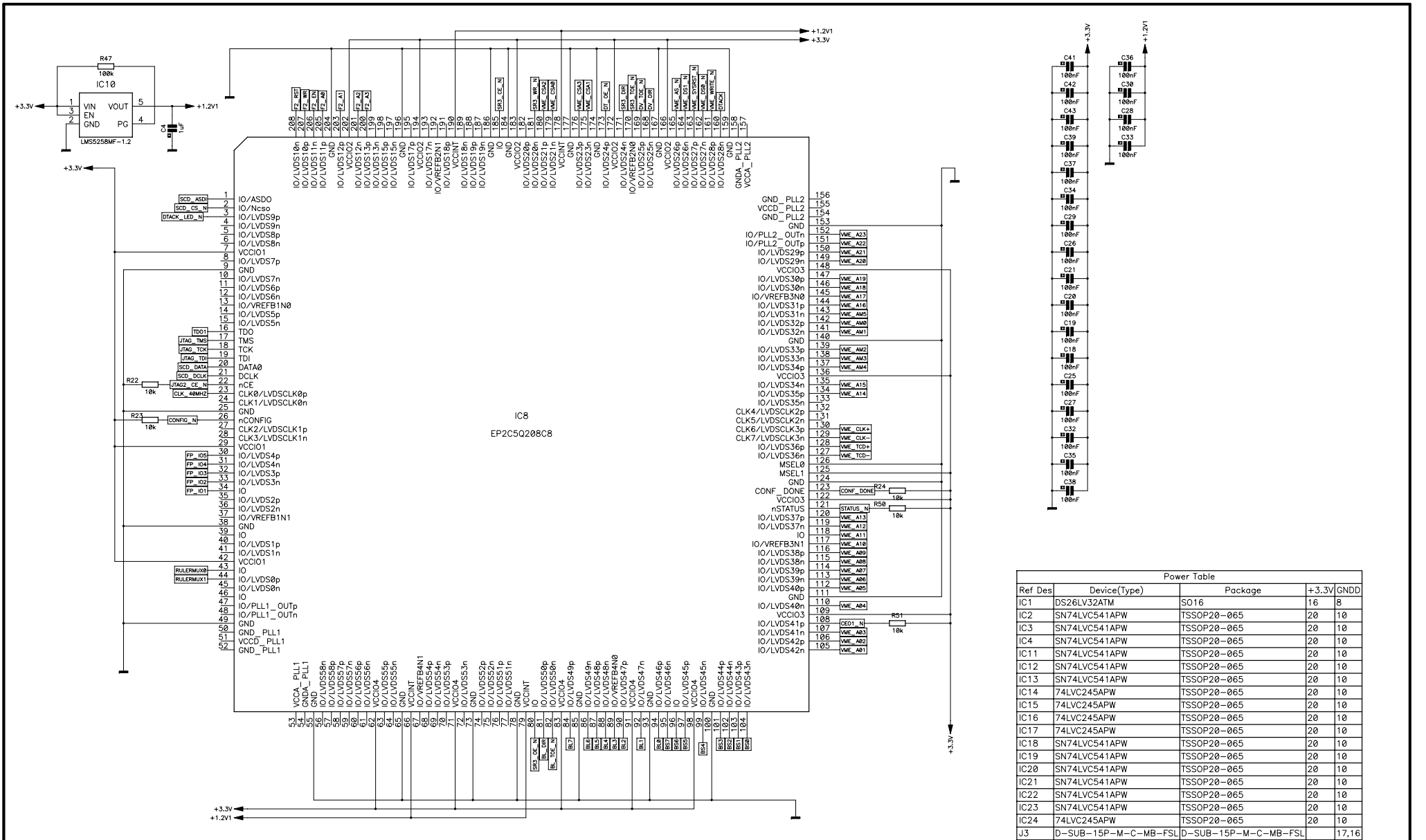
H.1. Schematics



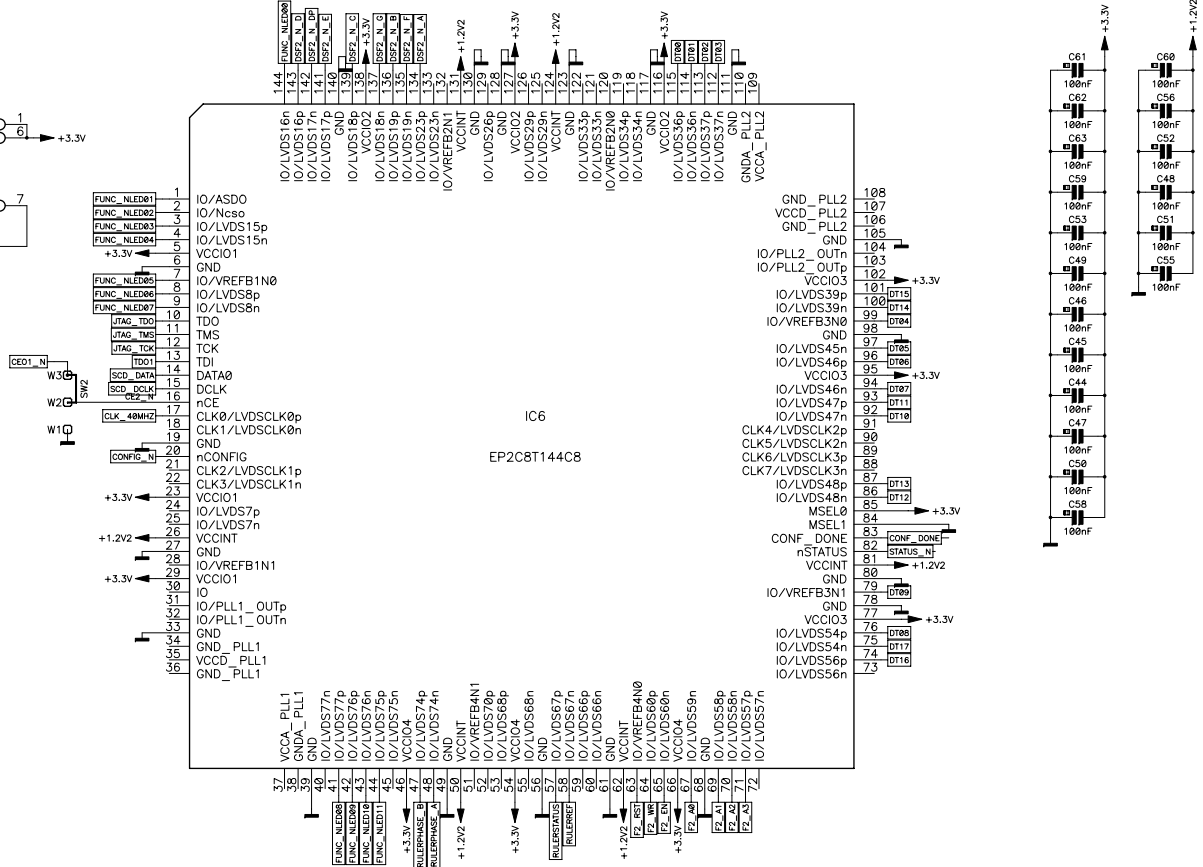
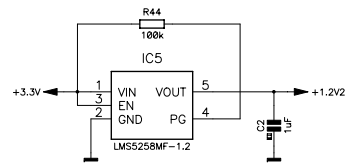
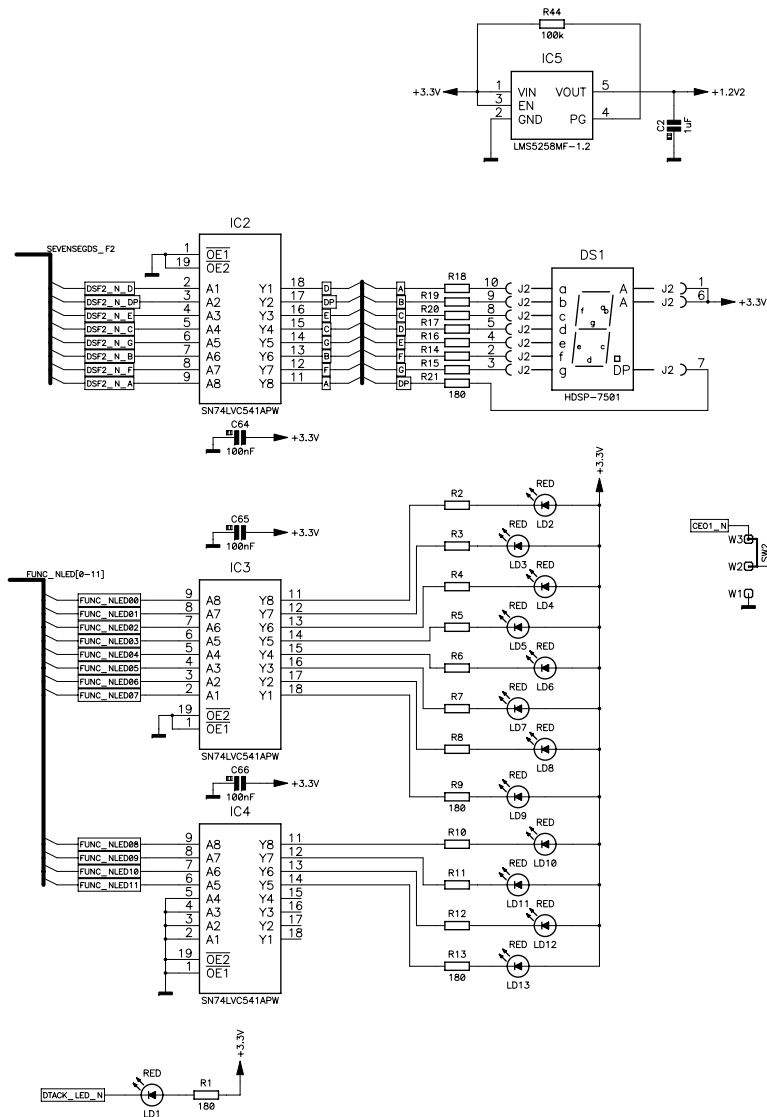
Proj/ Project:		Resp.: Koopman J.	74876
Ensemble/ Set:		Designer: Stian FØrde	
Sous-ensemble/ Underset:		Drawn: Stian FØrde	
Document:		MODA:	
FPGA prototype		MODB:	
Echelle/scale 1/1		File: EDA-01298-V1_sch.sch	INDICE
Sheet 1 of 4		EDA-01298	
LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENÈVE / GENEVA		Ref: nb	V1



Projet/Project: Ensemble/Set: Sous-ensemble/Underset: Document:		Resp.: Koopman J. 74876 Designer: Stian FØrde Drawn: Stian FØrde MODA: MODB:	
Eche/Scale: 1/1		File: EDA-01298-V1_sch.sch	
FPGA prototype Sheet 2 of 4		Ref: nb	
LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENEVA / GENEVA		EDA-01298 V1	

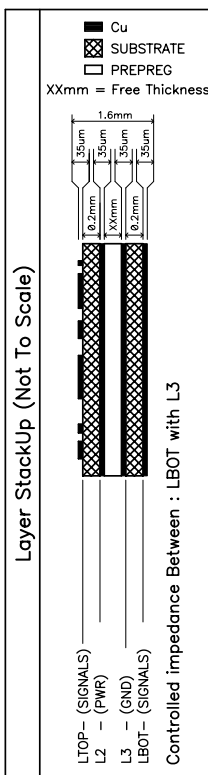
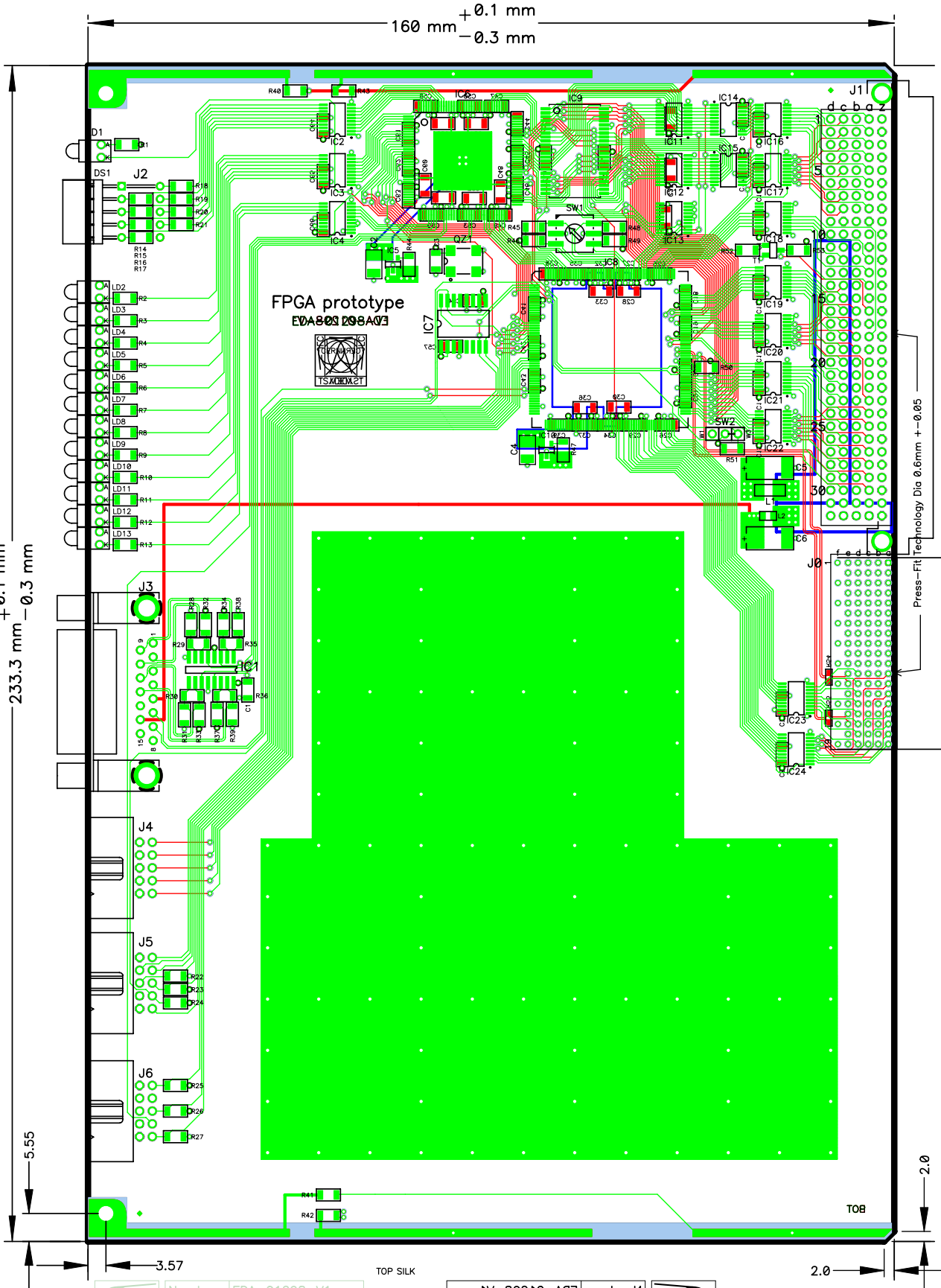


Projet/Project: Ensemble/Set: Sous-ensemble/Underset: Document:		Echelle/scale: 1/1	
		Responsible: Koopman J. 74876	
		Designer: Stian FPrde	
Document: FPGA prototype		Drawn: Stian FPrde	
		MODE:	
Sheet 3 of 4		File: EDA-01298-V1_sch.sch	
LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENEVE / GENEVA		Ref. nb	INDEXE EDA-01298 V1



Projet / Project:		Resp.: Koopman J.	74876
Ensemble / Set:		Designer: Stian FØrde	
Sous-ensemble / Underset:		Drawn: Stian FØrde	
Document:		MODA:	
FPGA prototype		MODB:	
Sheet 4 of 4		File: EDA-01298-V1_sch.sch	INDICE
LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENEVE / GENEVA		Ref: nb	EDA-01298 V1

H.2. PCB



Drill Table

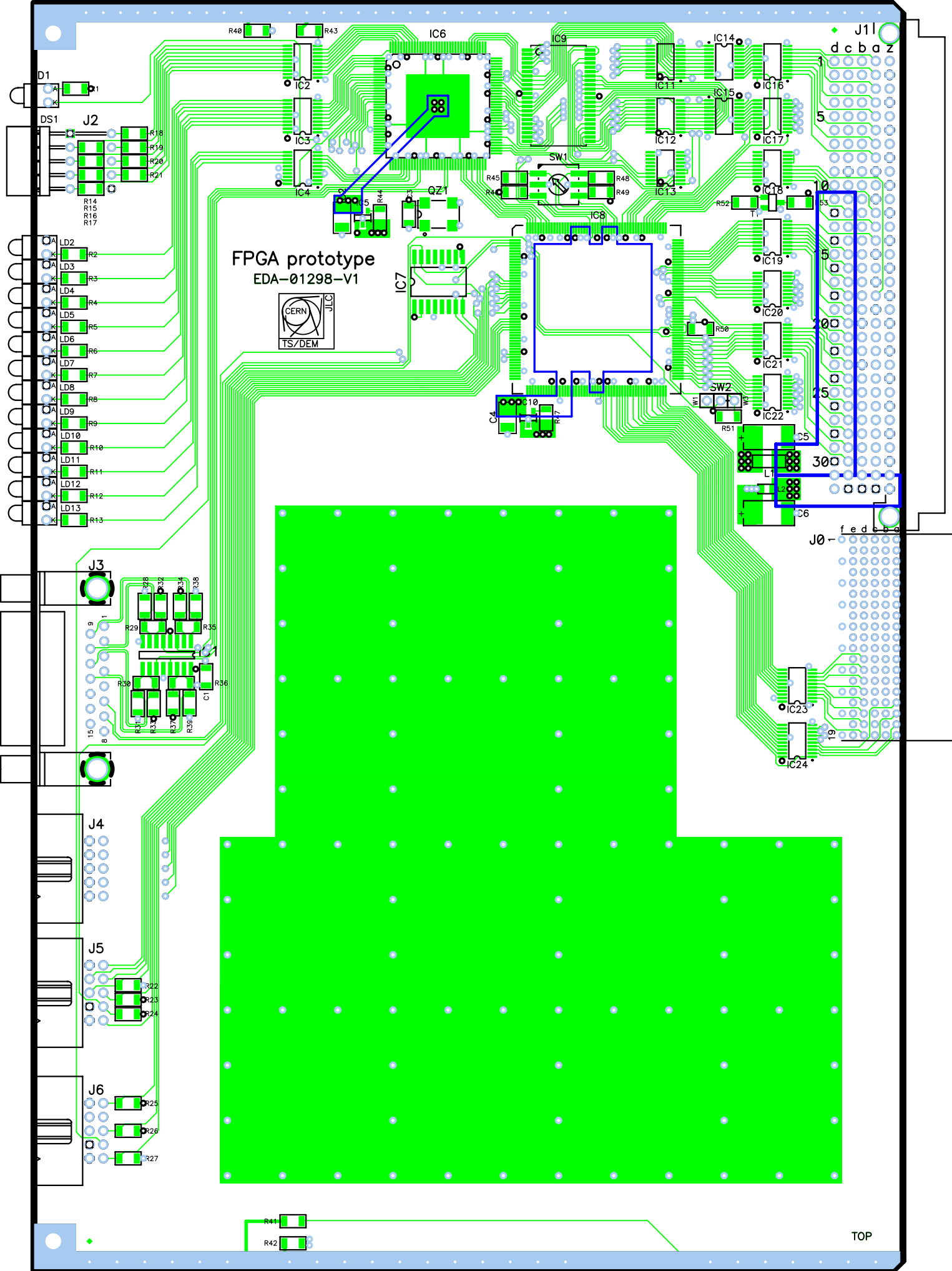
Hole Dia (mm)	Symbol	Quantity	Plated	Comments
0.400	+	467	Yes	
0.500	X	90	Yes	
0.600	Y	105	Yes	Press-Fit Technology Dia=0.6 +/- 0.05
0.800	T	10	Yes	
0.900	X	18	Yes	
1.000	W	216	Yes	Press-Fit Technology Dia=1.0 +/- 0.1
2.800	□	4	Yes	
3.200	◇	2	Yes	

CERN TS/DEM	Number	EDA-01298-V1
	Drawn By	CEGELEC JLC
	DATE	15/03/2006

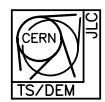
TOP SILK
TOP
BOTTOM
BOTTOM SILK

Number	EDA-01298-V1	CERN
Drawn By	CEGELEC JLC	
DATE	15/03/2006	TS/DEM





FPGA prototype
EDA-01298-V1



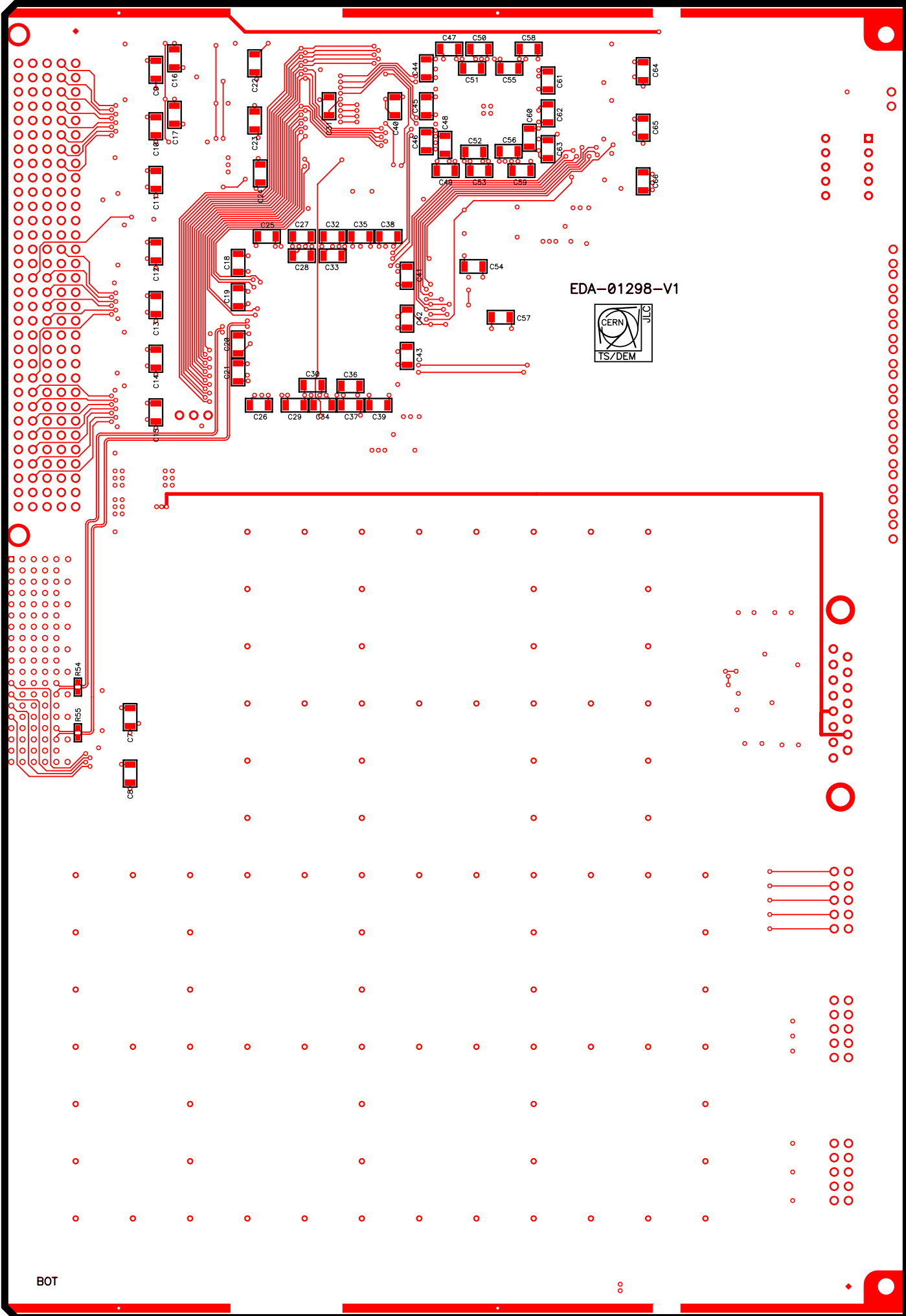
TOP

TOP SILK

TOP

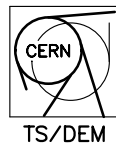


Number	EDA-01298-V1
Drawn By	CEGELEC JLC
DATE	15/03/2006



BOT

4



Number	EDA-01298-V1
Drawn By	CEGELEC JLC
DATE	15/03/2006

BOTTOM

BOTTOM SILK

H.3. FPGA pin locations

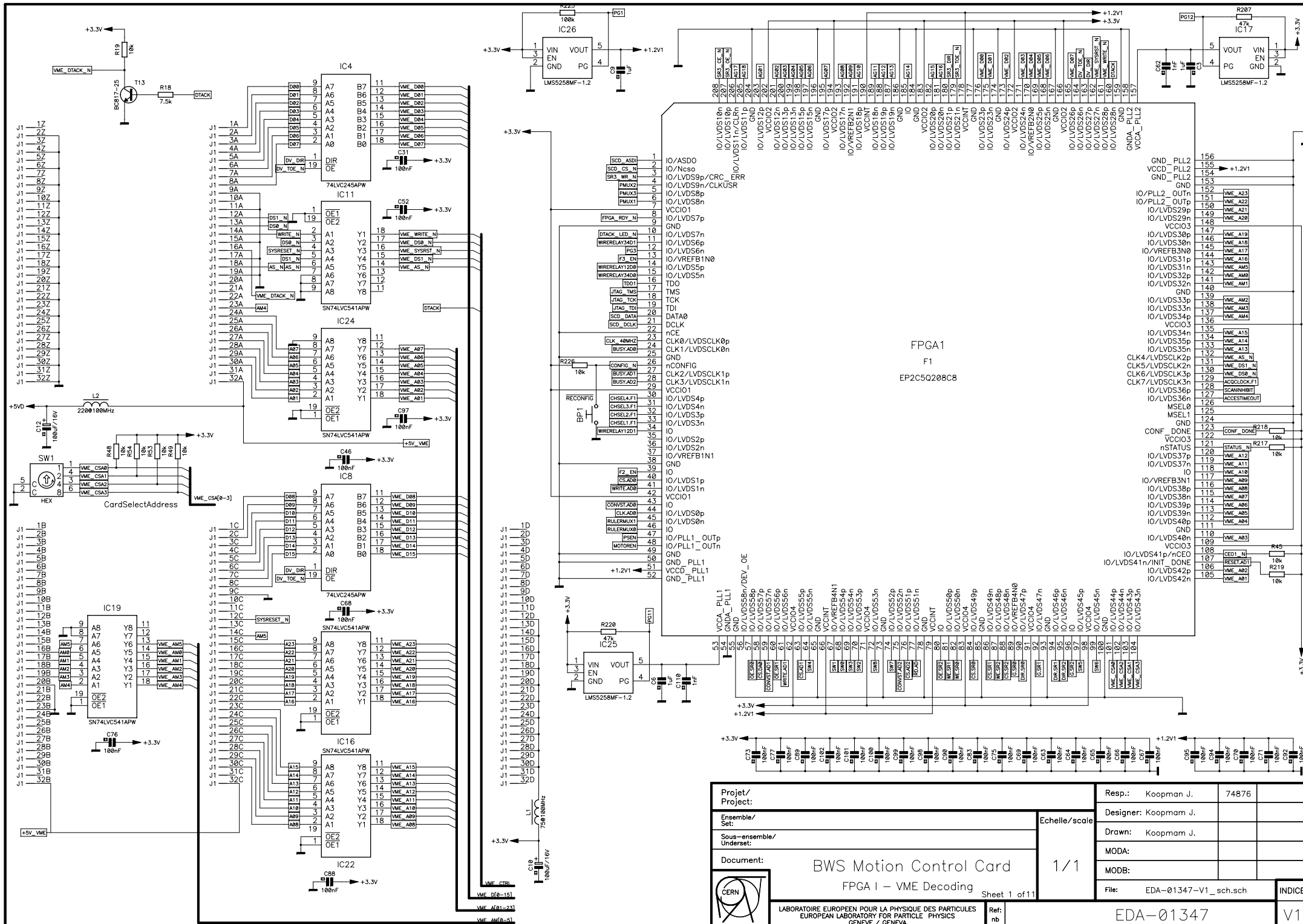
Master FPGA			
USER PIN LOCATION TABLE			
Pin name	Pin nr	Pin name	Pin nr
CLK_40MHz	23	BL0	94
		BL1	92
VME_SYSRST_N	163	BL2	90
VME_WRITE_N	161	BL3	89
VME_AS_N	165	BL4	88
VME_DS0_N	162	BL5	87
VME_DS1_N	164	BL6	86
VME_A01	105	BL7	84
VME_A02	106	BS0	104
VME_A03	107	BS1	103
VME_A04	110	BS2	102
VME_A05	112	BS3	101
VME_A06	113	BS4	99
VME_A07	114	BS5	97
VME_A08	115	BS6	96
VME_A09	116	BS7	95
VME_A10	117		
VME_A11	118	F2_A0	205
VME_A12	119	F2_A1	203
VME_A13	120	F2_A2	201
VME_A14	134	F2_A3	200
VME_A15	135	F2_EN	206
VME_A16	144	F2_WR	207
VME_A17	145	F2_RST	208
VME_A18	146		
VME_A19	147	FP_IO1	30
VME_A20	149	FP_IO2	31
VME_A21	150	FP_IO3	32
VME_A22	151	FP_IO4	33
VME_A23	152	FP_IO5	34
VME_AM0	142		
VME_AM1	141	BL_DIR	81
VME_AM2	139	BL_TOE_N	82
VME_AM3	138	DV_DIR	168
VME_AM4	137	DV_TOE_N	169
VME_AM5	143	SR3_OE_N	80
VME_CSA0	181	SR3_TOE_N	170
VME_CSA1	180	SR3_DIR	171
VME_CSA2	179	SR3_CE_N	185
VME_CSA3	175	SR3_WR_N	176
DTACK	160	DT_OE_N	173
DTACK_LED_N	133		
VME_TCD- (LVDSn)	127	RULERMUX0	43
VME_TCD+ (LVDSp)	128	RULERMUX1	44
VME_CLK- (LVDSn)	129		
VME_CLK+ (LVDSp)	130		

H.3. FPGA PIN LOCATIONS

Slave1 FPGA			
USER PIN LOCATION TABLE			
Pin name	Pin nr	Pin name	Pin nr
CLK_40MHz	17	RULERSTATUS	57
F2_RST	63	RULERREF	58
F2_WR	64	RULERPHASE_A	48
F2_EN	65	RULERPHASE_B	47
F2_A0	67	FUNC_NLED00	144
F2_A1	69	FUNC_NLED01	1
F2_A2	70	FUNC_NLED02	2
F2_A3	71	FUNC_NLED03	3
DT00	115	FUNC_NLED04	4
DT01	114	FUNC_NLED05	7
DT02	113	FUNC_NLED06	8
DT03	112	FUNC_NLED07	9
DT04	99	FUNC_NLED08	41
DT05	97	FUNC_NLED09	42
DT06	96	FUNC_NLED10	43
DT07	94	FUNC_NLED11	44
DT08	76	DSF2_N_A	134
DT09	79	DSF2_N_B	136
DT10	92	DSF2_N_C	139
DT11	93	DSF2_N_D	143
DT12	86	DSF2_N_E	141
DT13	87	DSF2_N_F	135
DT14	100	DSF2_N_G	137
DT15	101	DSF2_N_DP	142
DT16	74		
DT17	75		

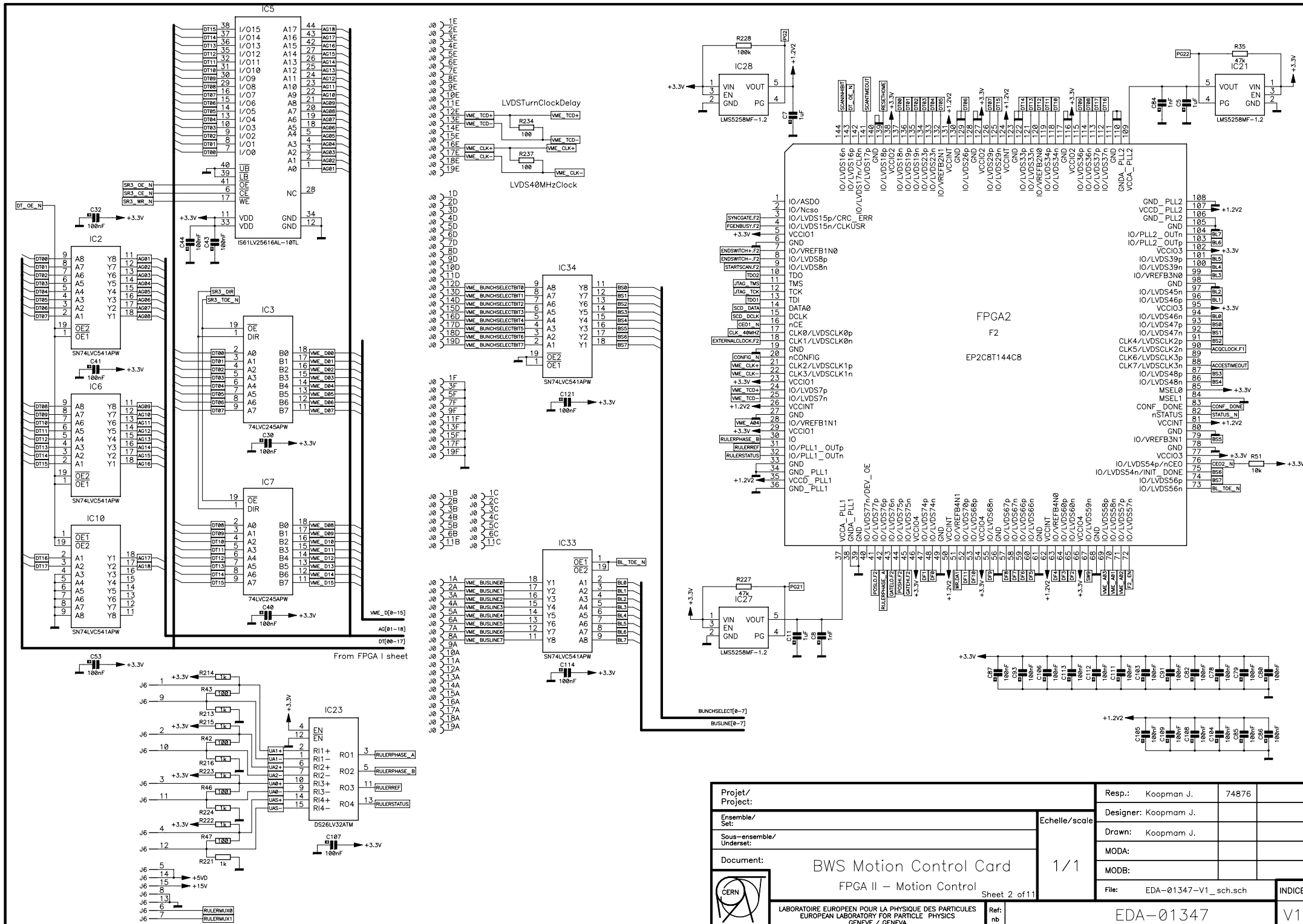
I. Prototype card

I.1. Schematics

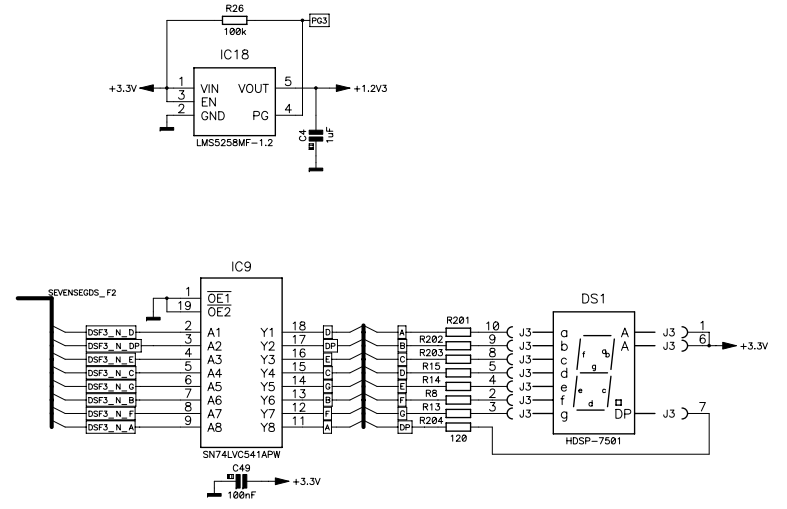
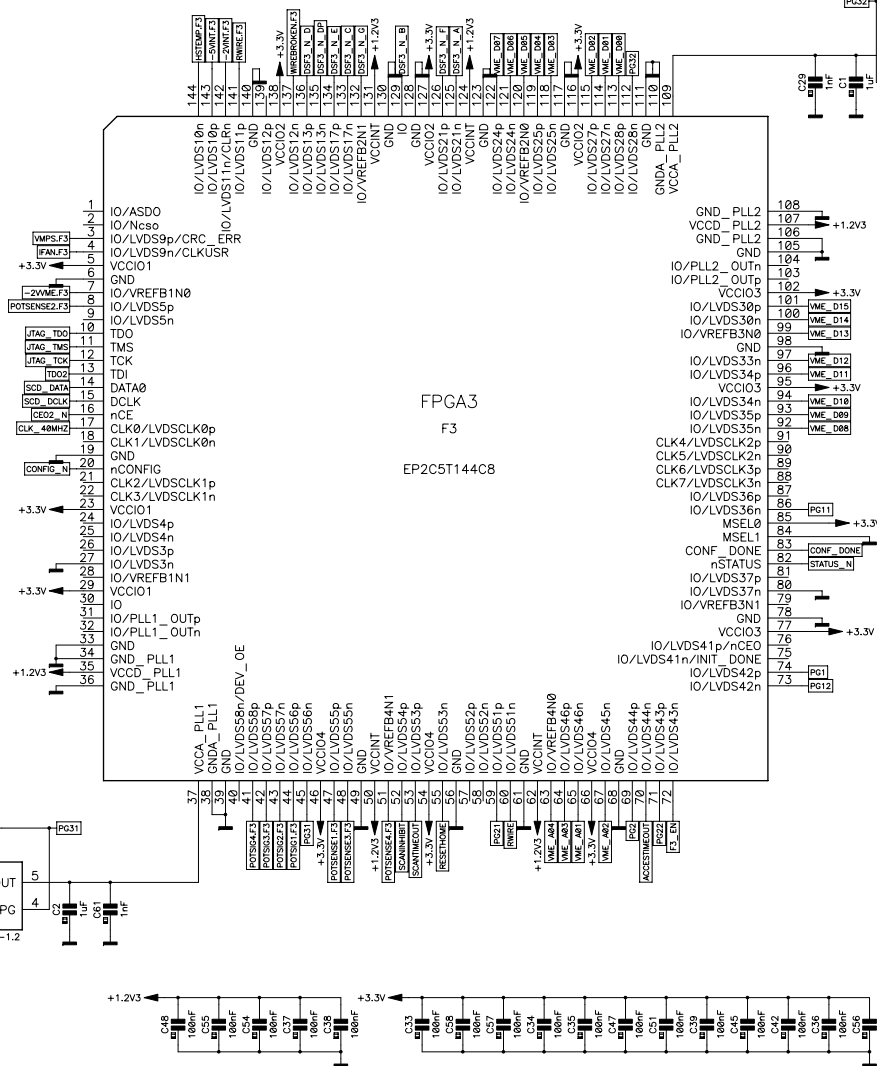


Proj/ Project:	Resp.: Koopman J.	74876
Ensemble/ Set:	Designer: Koopman J.	
Sous-ensemble/ Underset:	Drawn: Koopman J.	
Document:	MODA:	
	MODB:	
	File: EDA-01347-V1_sch.sch	INDICE

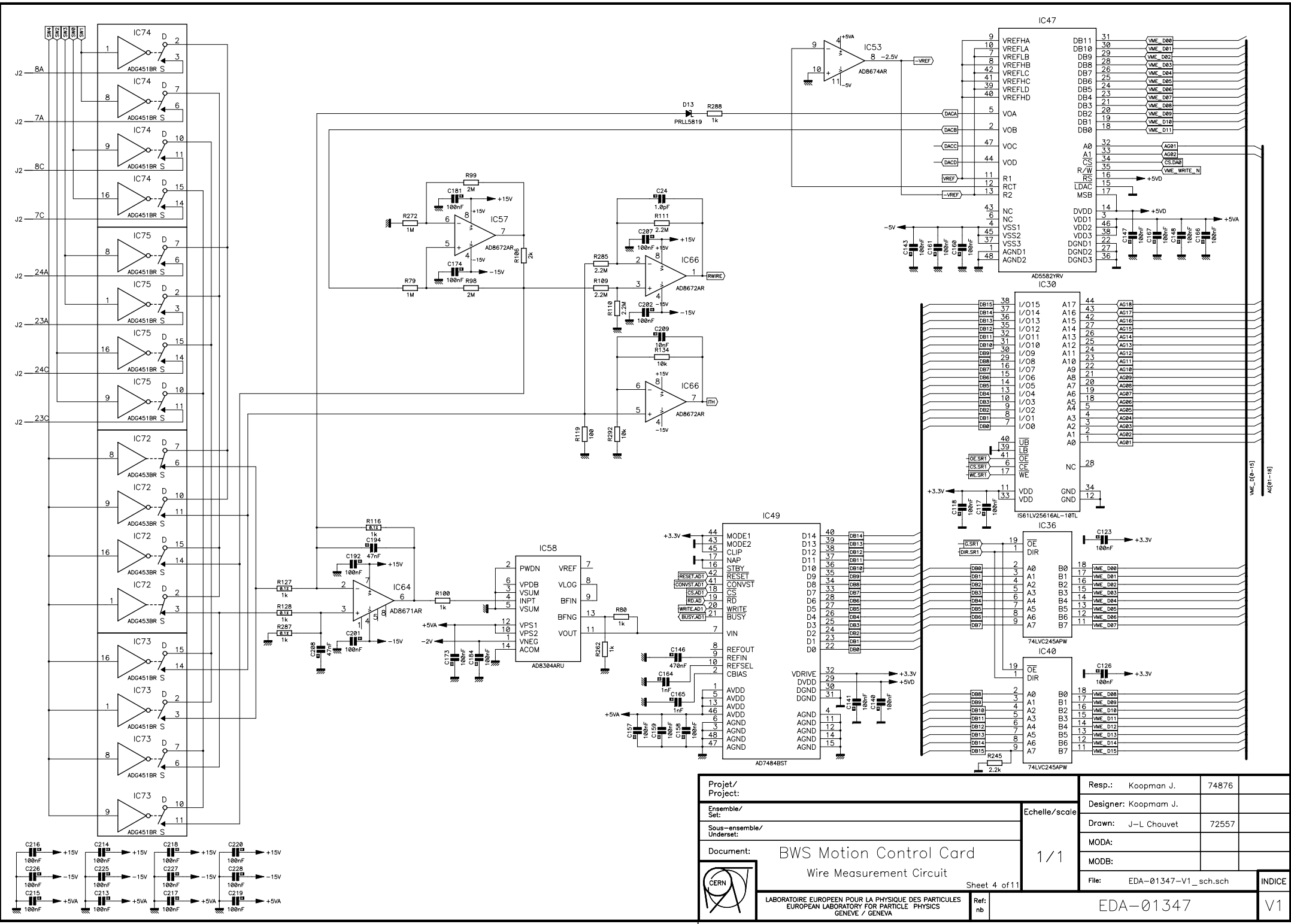
BWS Motion Control Card
FPGA I – VME Decoding
Sheet 1 of 11




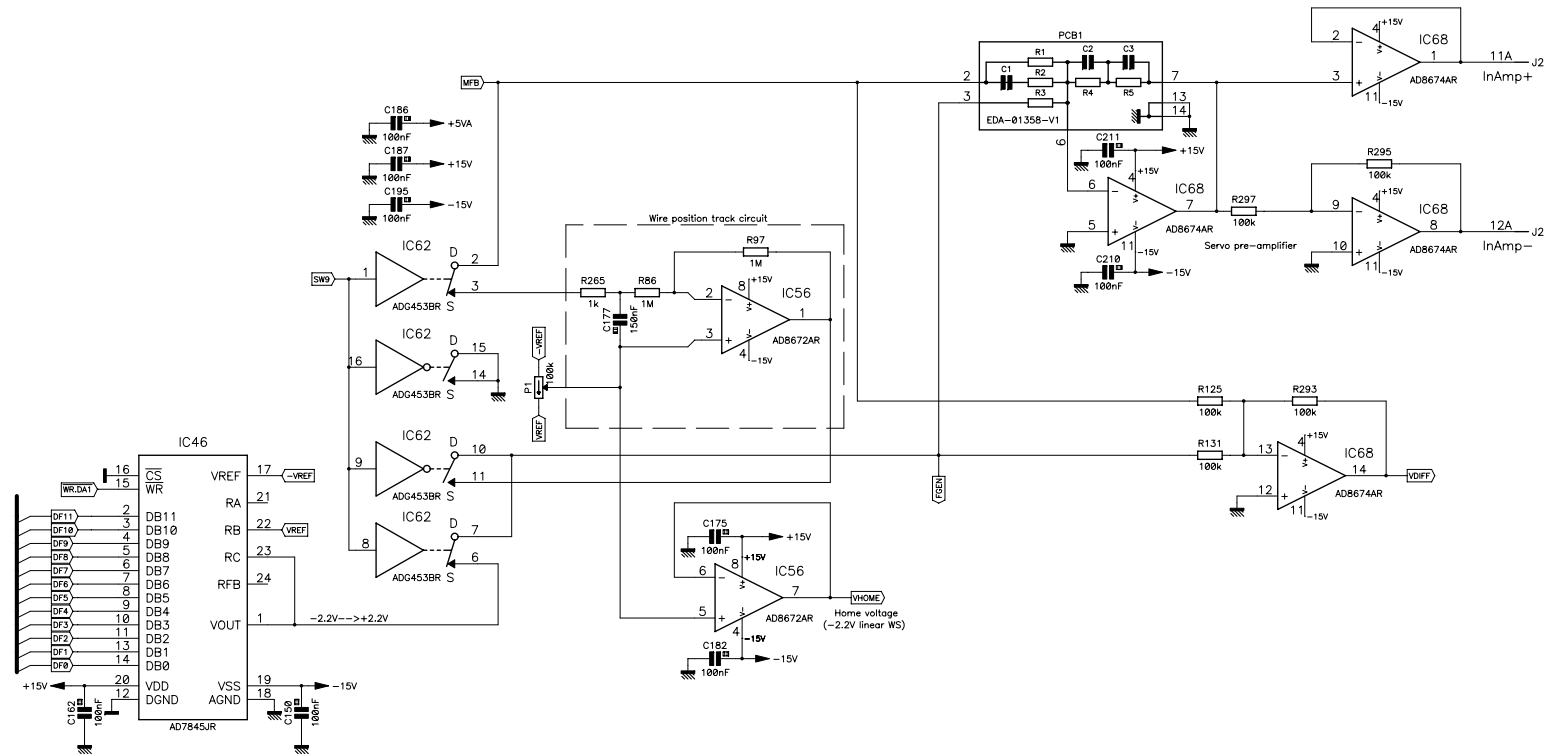
Projet/Project: BWS Motion Control Card		Resp.: Koopman J. 74876	
Ensemble/Set:		Designer: Koopman J.	
Sous-ensemble/Underset:		Drawn: Koopman J.	
Document:		MODA:	
FPGA II – Motion Control		MODB:	
Sheet 2 of 11		File: EDA-01347-v1_sch.sch	
LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS		Ref. nb	
		EDA-01347	
		INDICE	
		V1	




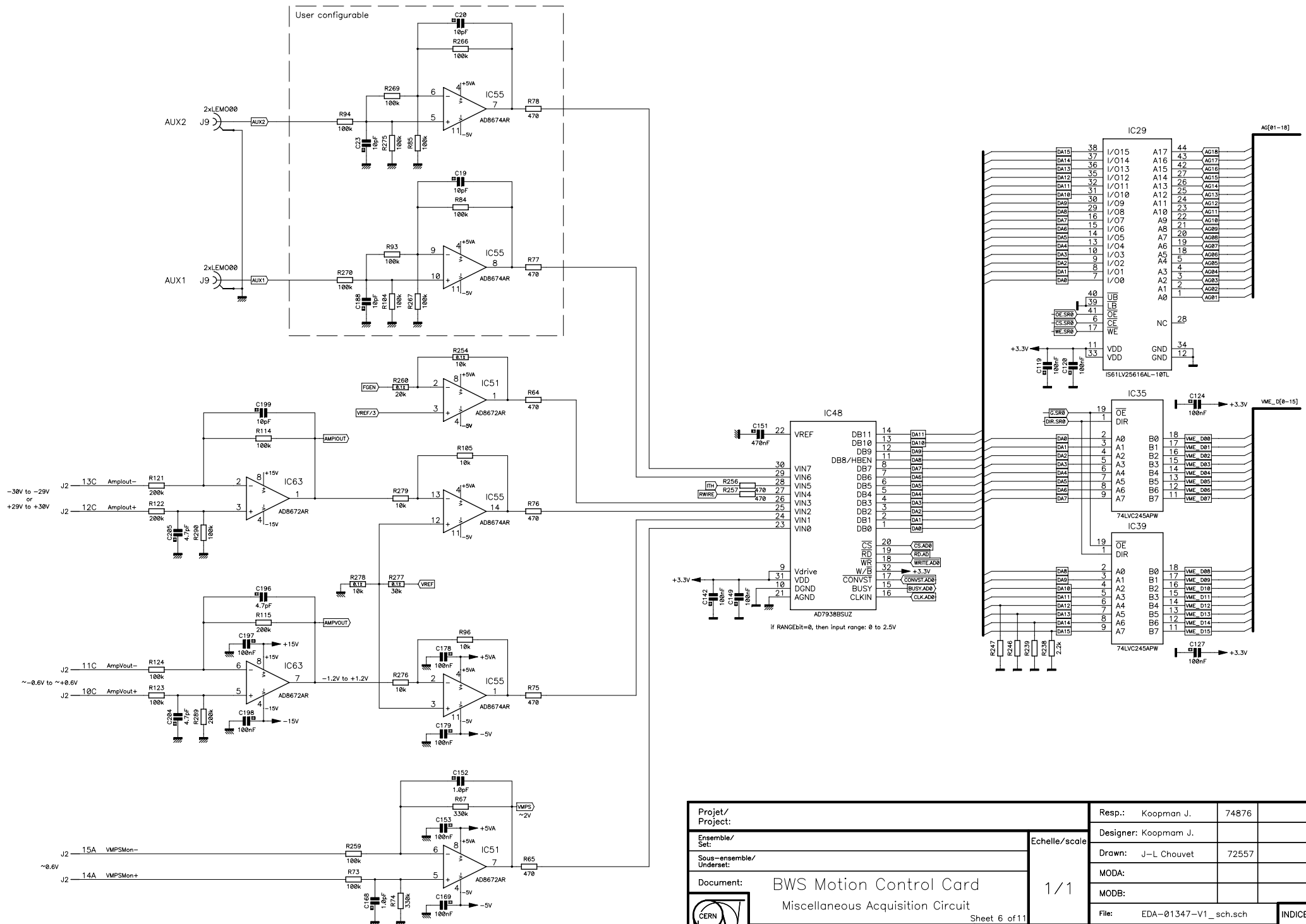
Proj/Project:		Resp.: Koopman J.	74876
Ensemble/Set:		Designer: Koopman J.	
Sous-ensemble/Underset:		Drawn: Koopman J.	
Document:		MODA:	
BWS Motion Control Card FPGA III - Status Handling		MODB:	
Sheet 3 of 11		File: EDA-01347-V1_sch.sch	INDICE
LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENEVA / GENEVA		Ref:	EDA-01347
		nb	V1



Proj/ Project:		
Ensemble/ Set:		
Sous-ensemble/ Underset:		
Document:	BWS Motion Control Card Wire Measurement Circuit	
	Sheet 4 of 11	Ref: nb
Echelle/scale	1/1	
Resp.:	Koopman J.	74876
Designer:	Koopman J.	
Drawn:	J-L Chouvet	72557
MODA:		
MODB:		
File:	EDA-01347-V1_sch.sch	
Ref:	EDA-01347	
INDICE	V1	

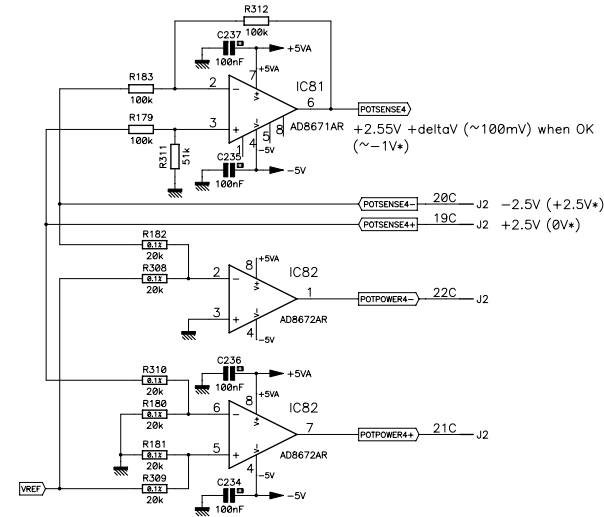
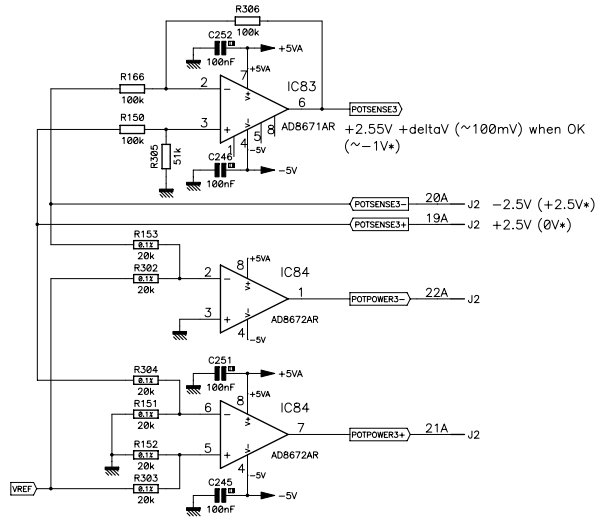
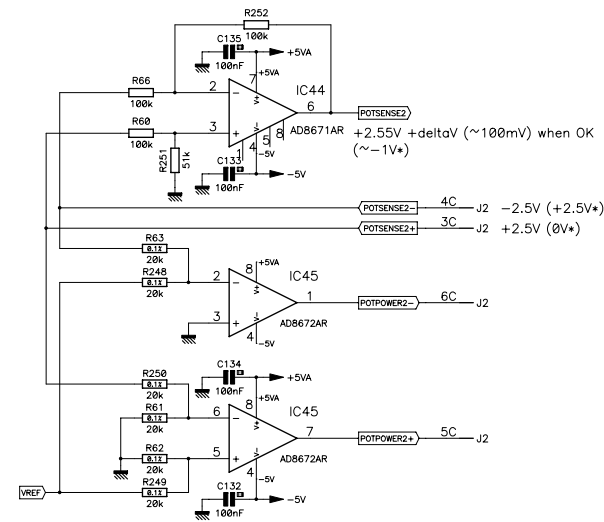
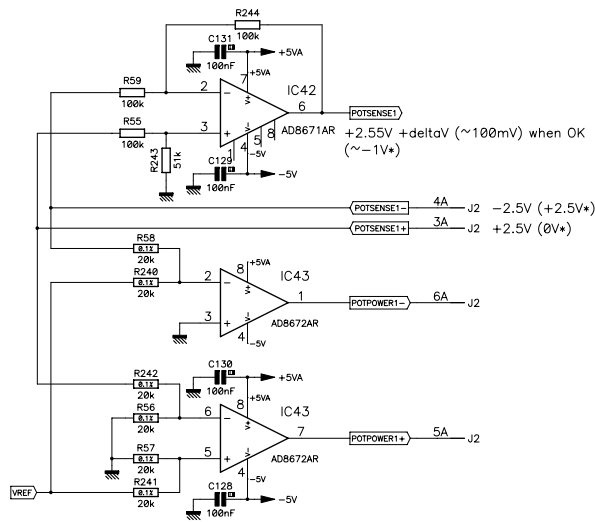


Projet/ Project:		Resp.: Koopman J.	74876	
Ensemble/ Set:		Designer: Koopman J.		
Sous-ensemble/ Underset:		Drawn: J-L Chauvet	72557	
Document: BWS Motion Control Card Function Generator Circuit		MODA:		
Echelle/scale: 1/1		MODB:		
Sheet 5 of 11		File: EDA-01347-V1_sch.sch		INDICE
 LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENEVE / GENEVA		Ref: nb	EDA-01347	V1

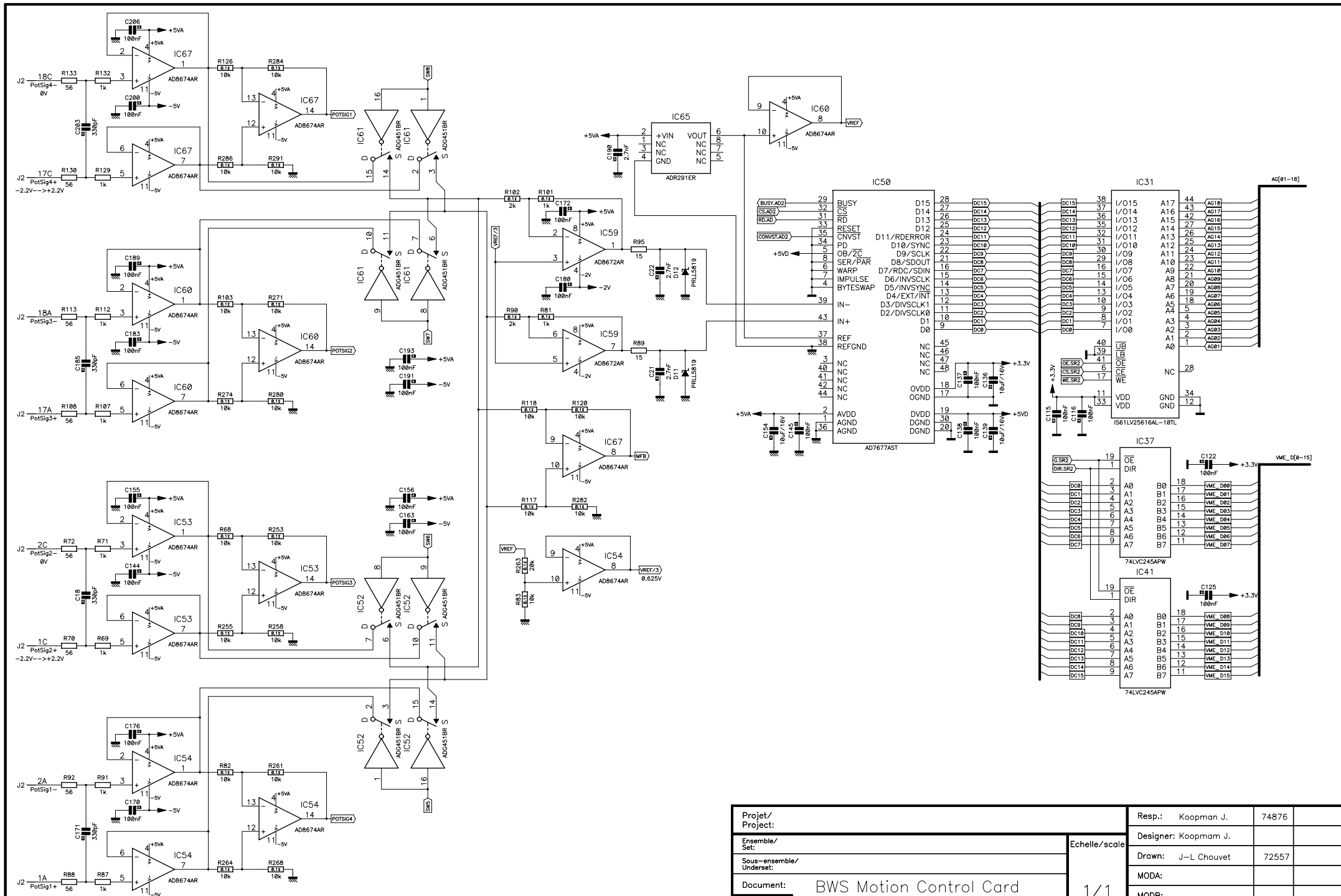


Projet / Project: Ensemble / Set: Sous-ensemble / Underset: Document:		Echelle / scale: 1 / 1		Resp.: Koopman J. 74876 Designer: Koopman J. Drawn: J-L Chauvet 72557 MODA: MODB:	
Project / Project: Ensemble / Set: Sous-ensemble / Underset: Document:		Echelle / scale: 1 / 1		File: EDA-01347-V1_sch.sch INDICE	
BWS Motion Control Card Miscellaneous Acquisition Circuit Sheet 6 of 11		LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENEVE / GENEVA		Ref: EDA-01347 nb: V1	

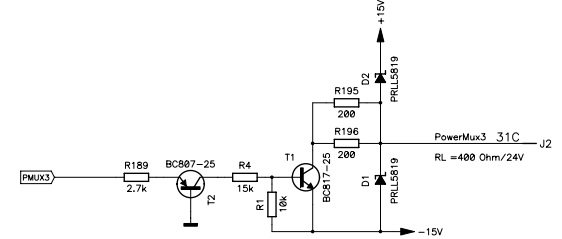
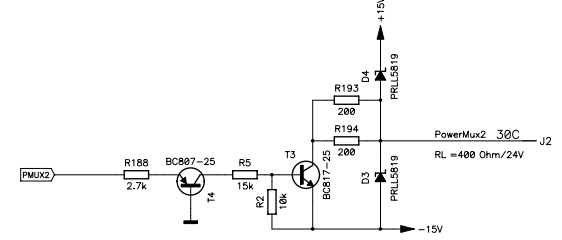
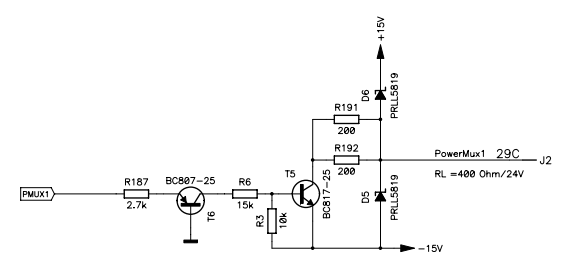
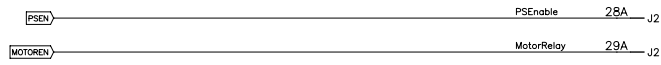
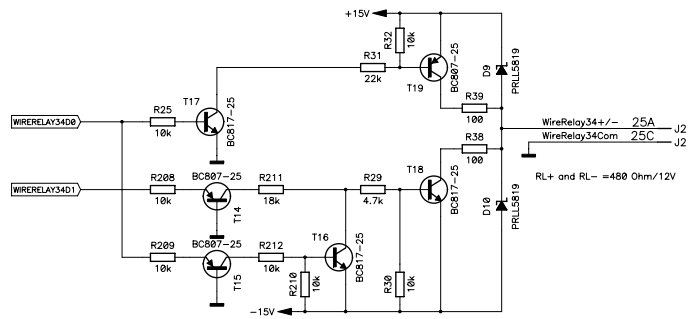
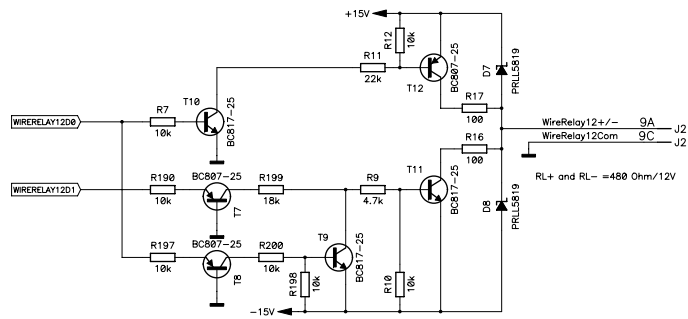
(* : when cable disconnected)



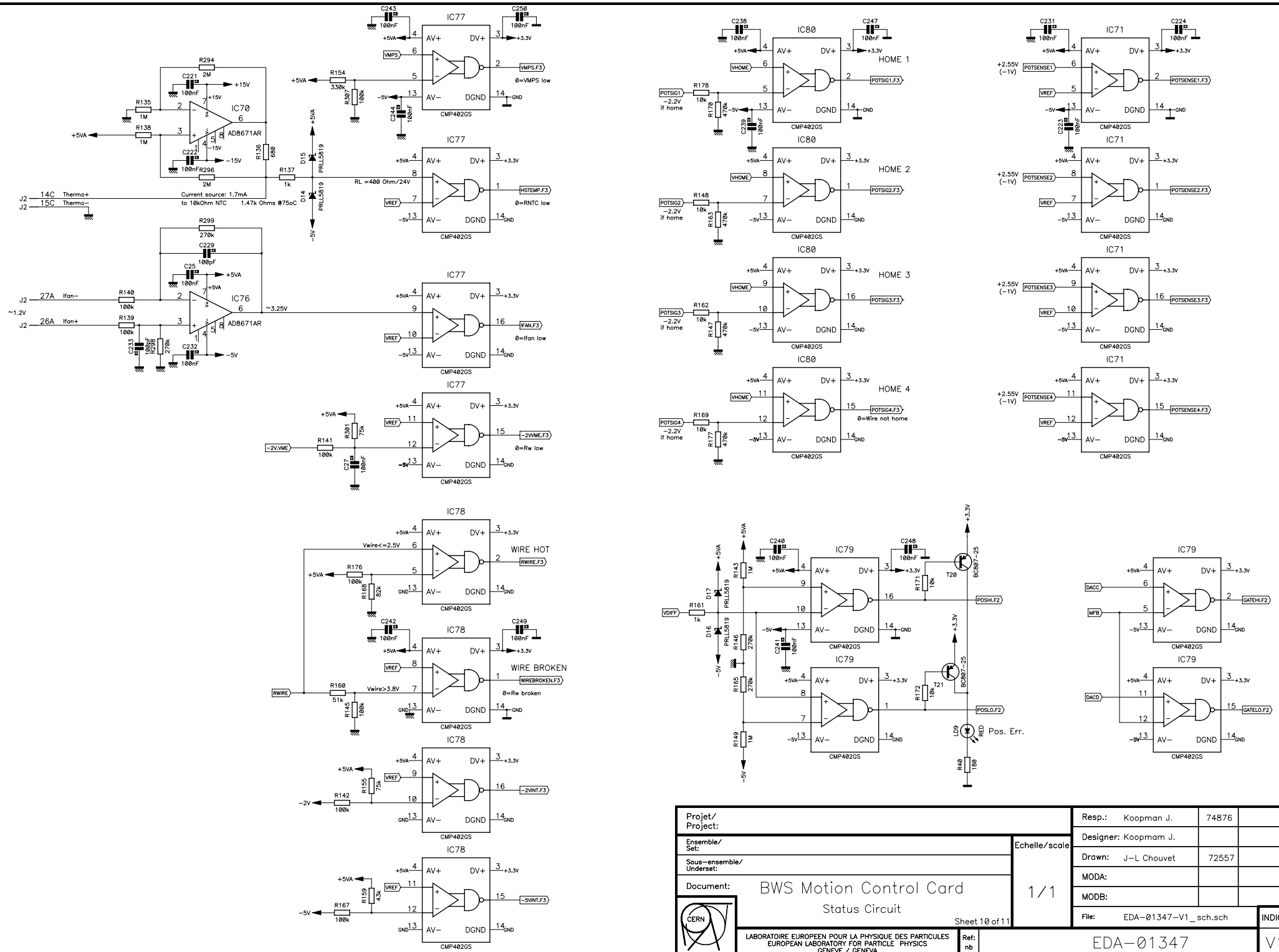
Projet/ Project:		Resp.: Koopman J.	74876
Ensemble/ Set:		Designer: Koopman J.	
Sous-ensemble/ Underset:		Drawn: J-L Chauvet	72557
Document: BWS Motion Control Card Potentiometer Powering circuit		MODA:	
Echelle/scale: 1/1		MODB:	
Sheet 7 of 11		File: EDA-01347-V1_sch.sch	INDICE
LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENEVE / GENEVA		Ref: nb	EDA-01347 V1



Proj / Project: Ensemble / Set: Sous-ensemble / Underset: Document:		BWS Motion Control Card Potentiometer Measurement circuit Sheet 8 of 11		Echelle/scale: 1/1		Resp.: Koopman J. 74876 Designer: Koopman J. Drawn: J-L Chauvet 72557 MODA: MODB:		File: EDA-01347-V1_sch.sch		INDICE V1	
LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENEVE / GENEVA		Ref: nb		EDA-01347							

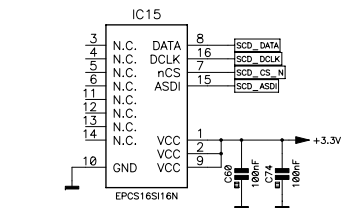
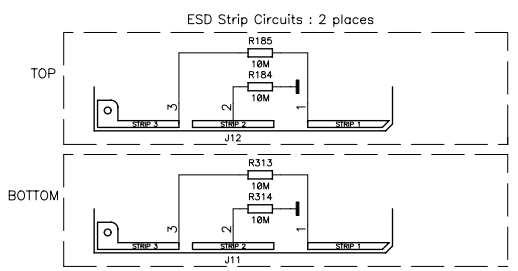
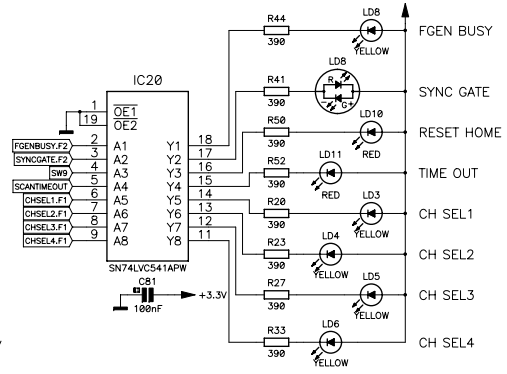
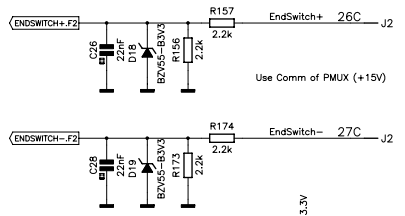
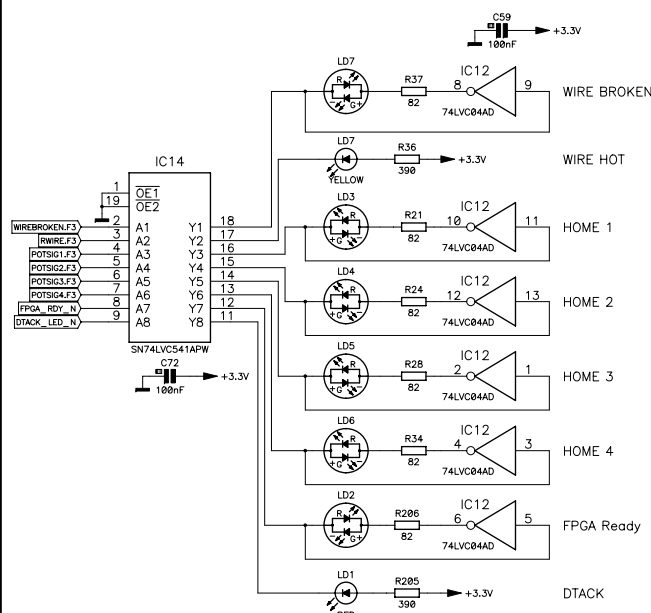
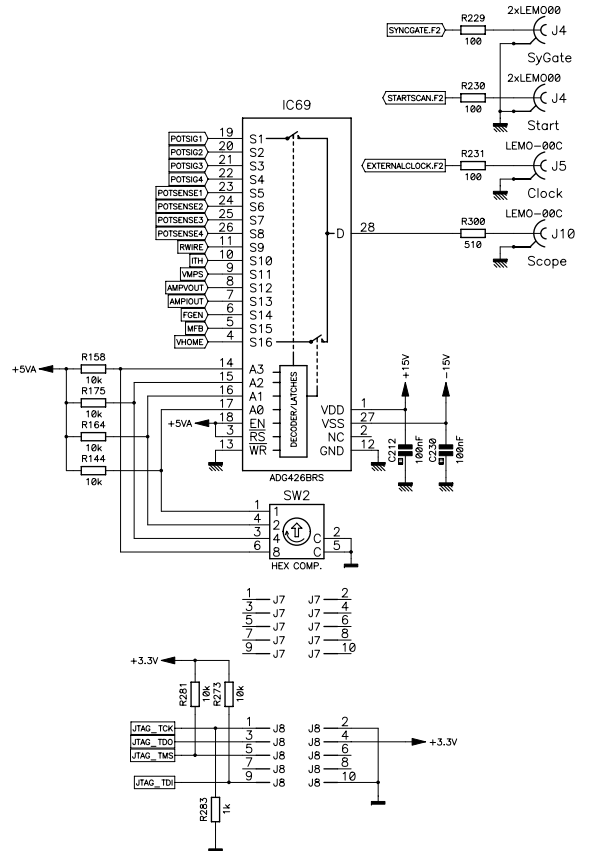
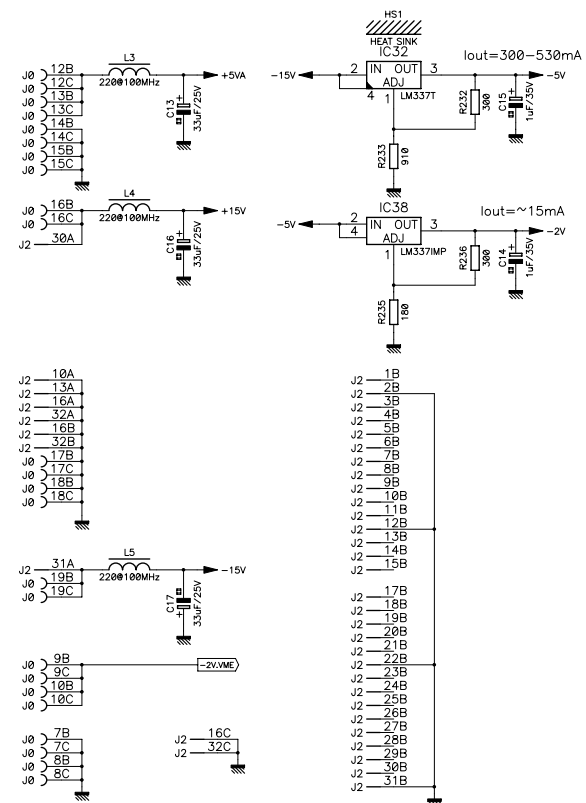


Projet/ Project:		Resp.: Koopman J.	74876
Ensemble/ Set:		Designer: Koopman J.	
Sous-ensemble/ Underset:		Drawn: J-L Chouvet	72557
Document:		MODA:	
BWS Motion Control Card Relay Control Circuit		MODB:	
Sheet 9 of 11		File: EDA-01347-V1_sch.sch	INDICE
		EDA-01347	
LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENEVE / GENEVA		Ref: nb	V1



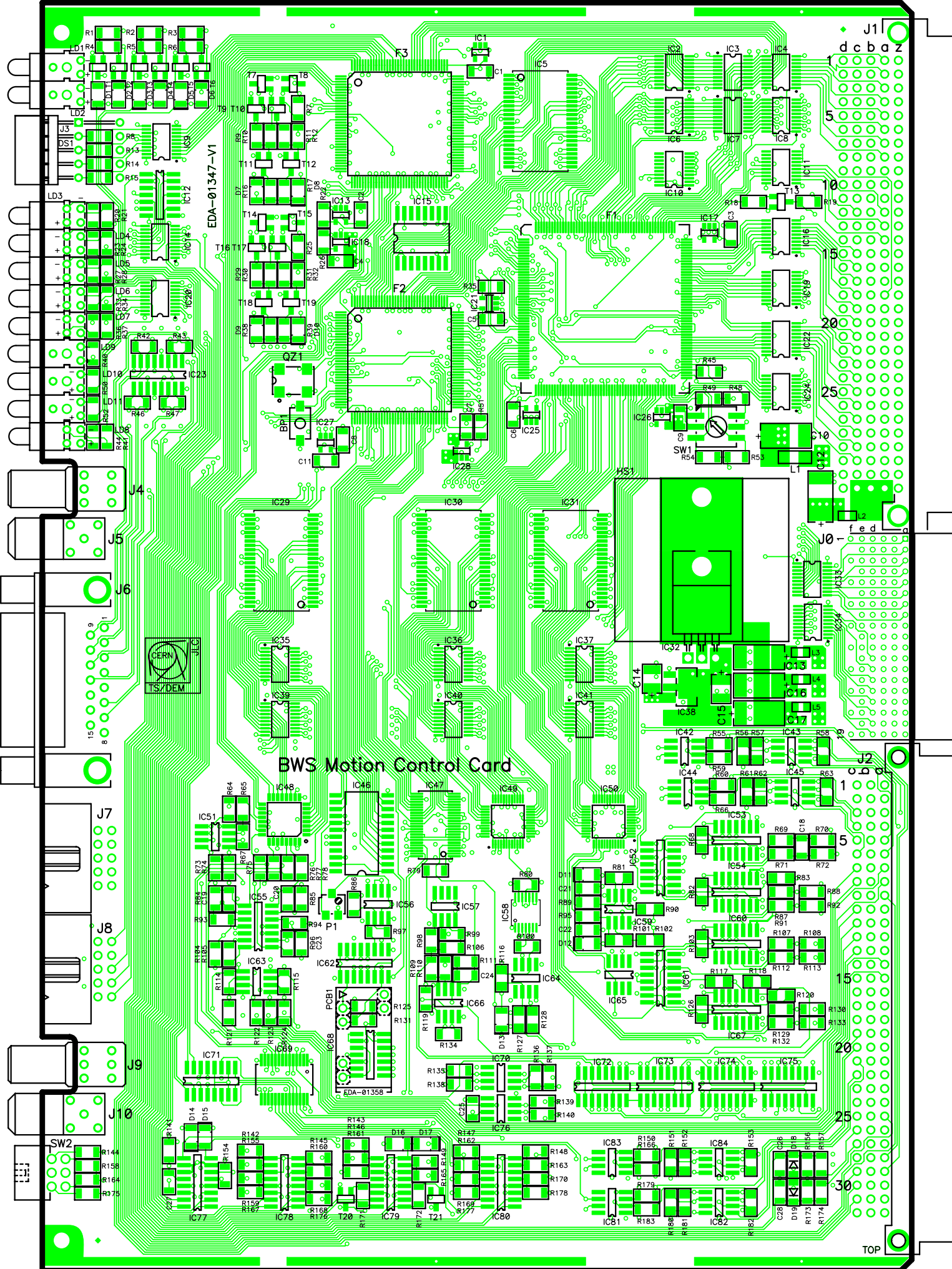
Projet/Project: Ensemble/Set: Sous-ensemble/Underset: Document:		Echelle/scale: 1/1	
BWS Motion Control Card Status Circuit		Sheet 10 of 11	
		LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENEVE / GENEVA	
Ref: nb		Resp.: Koopman J. 74876 Designer: Koopman J. Drawn: J-L Chouvet 72557 MODA: MODB: File: EDA-01347-V1_sch.sch	
		INDICE EDA-01347 V1	

Power Table									
Ref Des	Device(Type)	Package	+3.3V	GND	+15V	-15V	+5VA	-5V	
IC2	SN74LVC541APW	TSSOP20-065	20	10					
IC3	74LVC245APW	TSSOP20-065	20	10					
IC4	74LVC245APW	TSSOP20-065	20	10					
IC6	SN74LVC541APW	TSSOP20-065	20	10					
IC7	74LVC245APW	TSSOP20-065	20	10					
IC8	74LVC245APW	TSSOP20-065	20	10					
IC9	SN74LVC541APW	TSSOP20-065	20	10					
IC10	SN74LVC541APW	TSSOP20-065	20	10					
IC11	SN74LVC541APW	TSSOP20-065	20	10					
IC12	74LVC04AD	SO14	14	7					
IC14	SN74LVC541APW	TSSOP20-065	20	10					
IC16	SN74LVC541APW	TSSOP20-065	20	10					
IC19	SN74LVC541APW	TSSOP20-065	20	10					
IC20	SN74LVC541APW	TSSOP20-065	20	10					
IC22	SN74LVC541APW	TSSOP20-065	20	10					
IC23	DS26LV32ATM	SO16	16	8					
IC24	SN74LVC541APW	TSSOP20-065	20	10					
IC33	SN74LVC541APW	TSSOP20-065	20	10					
IC34	SN74LVC541APW	TSSOP20-065	20	10					
IC35	74LVC245APW	TSSOP20-065	20	10					
IC36	74LVC245APW	TSSOP20-065	20	10					
IC37	74LVC245APW	TSSOP20-065	20	10					
IC39	74LVC245APW	TSSOP20-065	20	10					
IC40	74LVC245APW	TSSOP20-065	20	10					
IC41	74LVC245APW	TSSOP20-065	20	10					
IC52	ADG451BR	SO16	5				13,12	4	
IC61	ADG451BR	SO16	5				13,12	4	
IC62	ADG453BR	SO16	5		13	4	12		
IC72	ADG453BR	SO16	5		13	4	12		
IC73	ADG451BR	SO16	5		13	4	12		
IC74	ADG451BR	SO16	5		13	4	12		
IC75	ADG451BR	SO16	5		13	4	12		
J6	D-SUB-15P-M-C-MB-FSL	D-SUB-15P-M-C-MB-FSL	17,16						



Proj/Project: BWS Motion Control Card Miscellaneous Circuits		Resp.: Koopman J. 74876	
Ensemble/Set:		Designer: Koopman J.	
Sous-ensemble/Underset:		Drawn: Koopman J.	
Document:		MODA:	
Echelle/scale: 1/1		MODB:	
Sheet 11 of 11		File: EDA-01347-V1_sch.sch	
		INDICE	
LABORATOIRE EUROPEEN POUR LA PHYSIQUE DES PARTICULES EUROPEAN LABORATORY FOR PARTICLE PHYSICS GENEVE / GENEVA		EDA-01347	
Ref. nb		V1	

I.2. PCB



BWS Motion Control Card

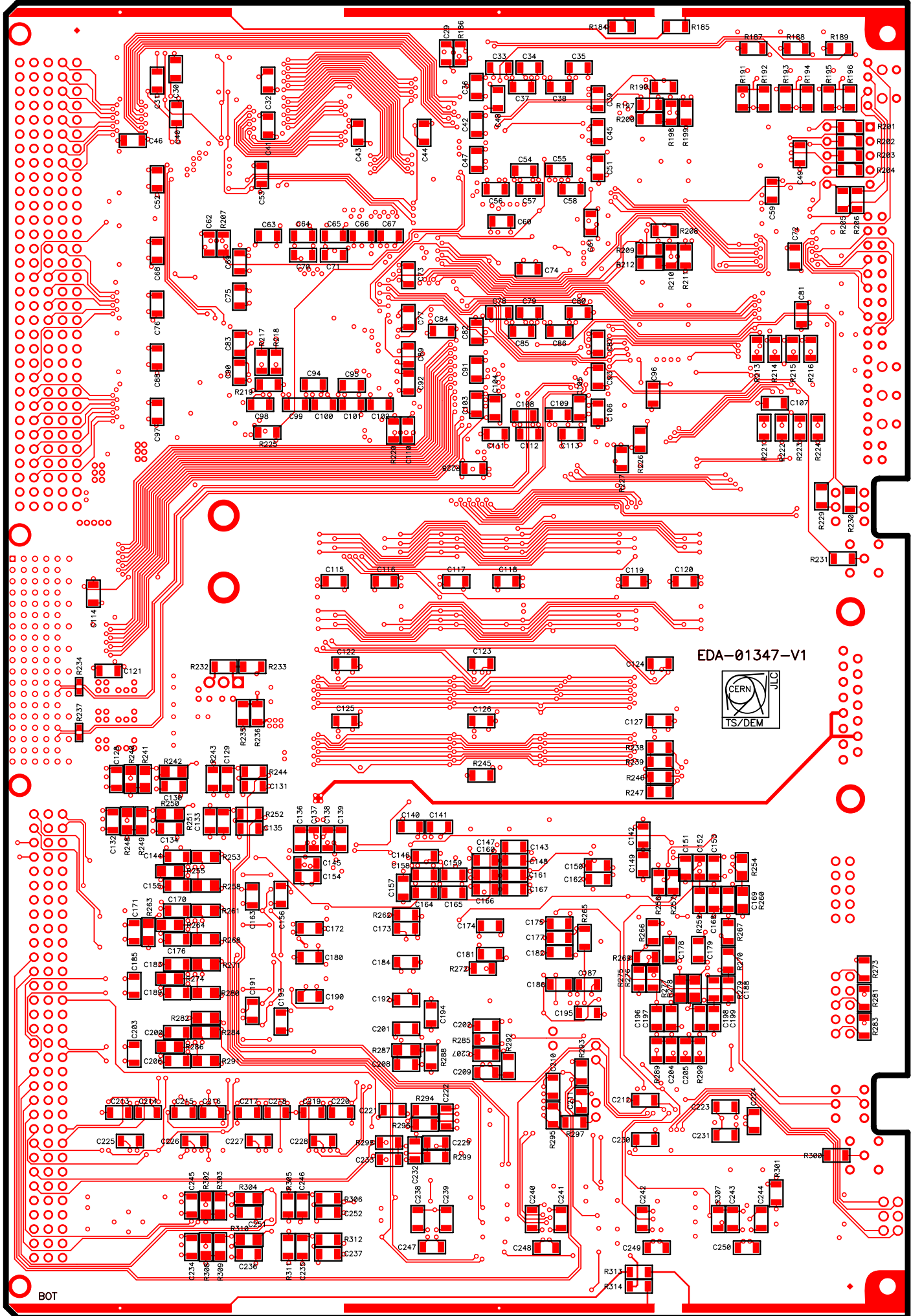


TOP SILK

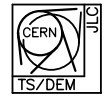
TOP



Number	EDA-01347-V1
Drawn By	CEGELEC JLC
DATE	15/04/2006



EDA-01347-V1



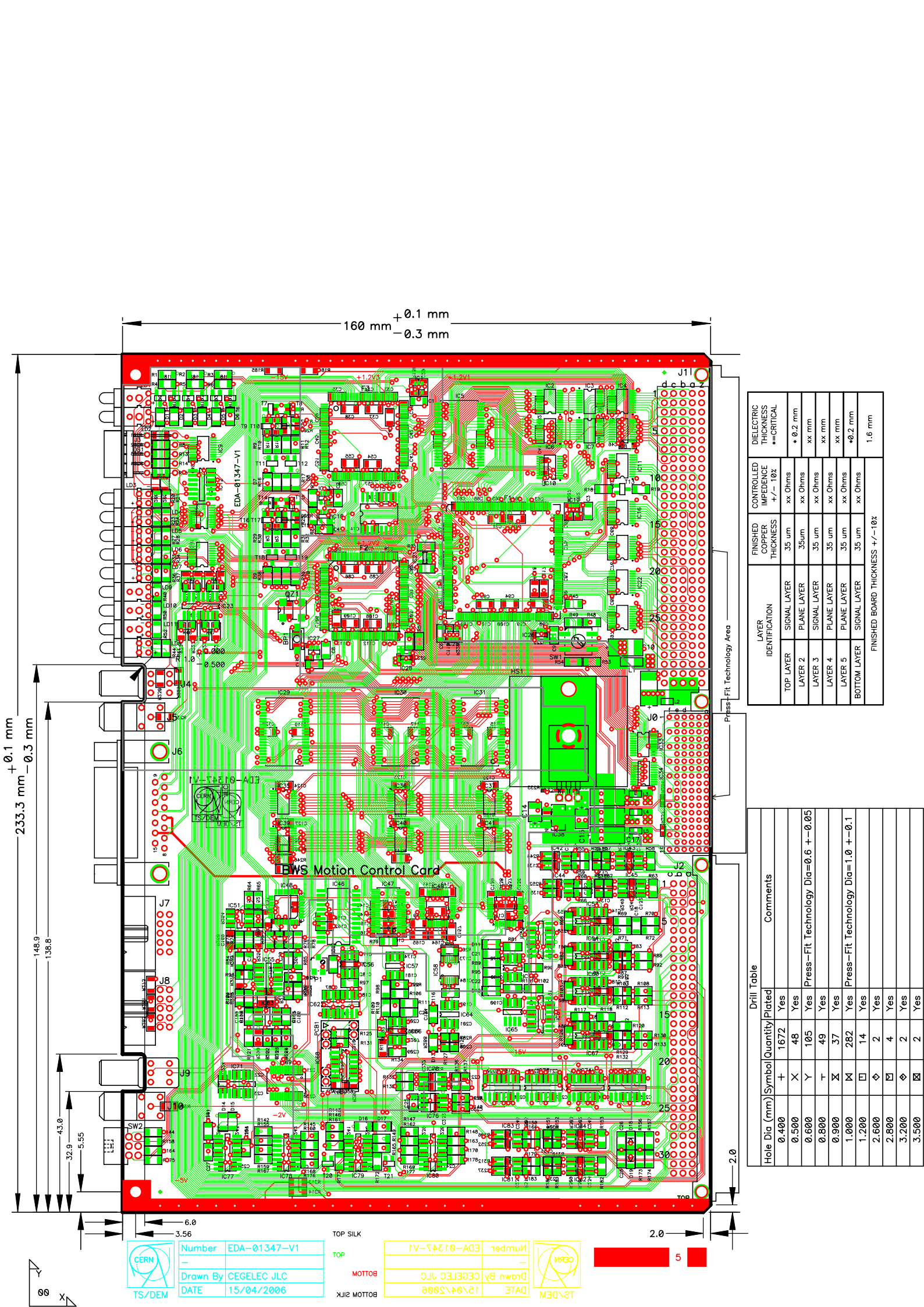
BOT



Number	EDA-01347-V1
Drawn By	CEGELEC JLC
DATE	15/04/2006

BOTTOM

BOTTOM SILK



160 mm $+0.1$ mm
 -0.3 mm

$+0.1$ mm
233.3 mm -0.3 mm

148.9
138.8
32.9
4.3.0
5.55
6.0
3.56

Press-Fit Technology Area

LAYER IDENTIFICATION	FINISHED COPPER THICKNESS	CONTROLLED IMPEDANCE +/- 10%	DIELECTRIC THICKNESS
TOP LAYER	SIGNAL LAYER	xx Ohms	* 0.2 mm
LAYER 2	PLANE LAYER	xx Ohms	xx mm
LAYER 3	SIGNAL LAYER	xx Ohms	xx mm
LAYER 4	PLANE LAYER	xx Ohms	xx mm
LAYER 5	PLANE LAYER	xx Ohms	* 0.2 mm
BOTTOM LAYER	SIGNAL LAYER	xx Ohms	1.6 mm
FINISHED BOARD THICKNESS +/- 10%			

Hole Dia (mm)	Symbol	Quantity	Plated	Comments
0.400	+	1672	Yes	
0.500	X	48	Yes	
0.600	Y	105	Yes	Press-Fit Technology Dia=0.6 + -0.05
0.800	T	49	Yes	
0.900	X	37	Yes	
1.000	M	282	Yes	Press-Fit Technology Dia=1.0 + -0.1
1.200	□	14	Yes	
2.600	◇	2	Yes	
2.800	◇	4	Yes	
3.200	◇	2	Yes	
3.500	⊠	2	Yes	

EWS Motion Control Card

CERN TS/DEM	Number	EDA-01347-V1
	Drawn By	CEGELEC JLC
	DATE	15/04/2006

CERN TS/DEM	Number	EDA-01347-V1
	Drawn By	CEGELEC JLC
	DATE	15/04/2006



TOP SILK
TOP
BOTTOM
BOTTOM SILK

2.0

