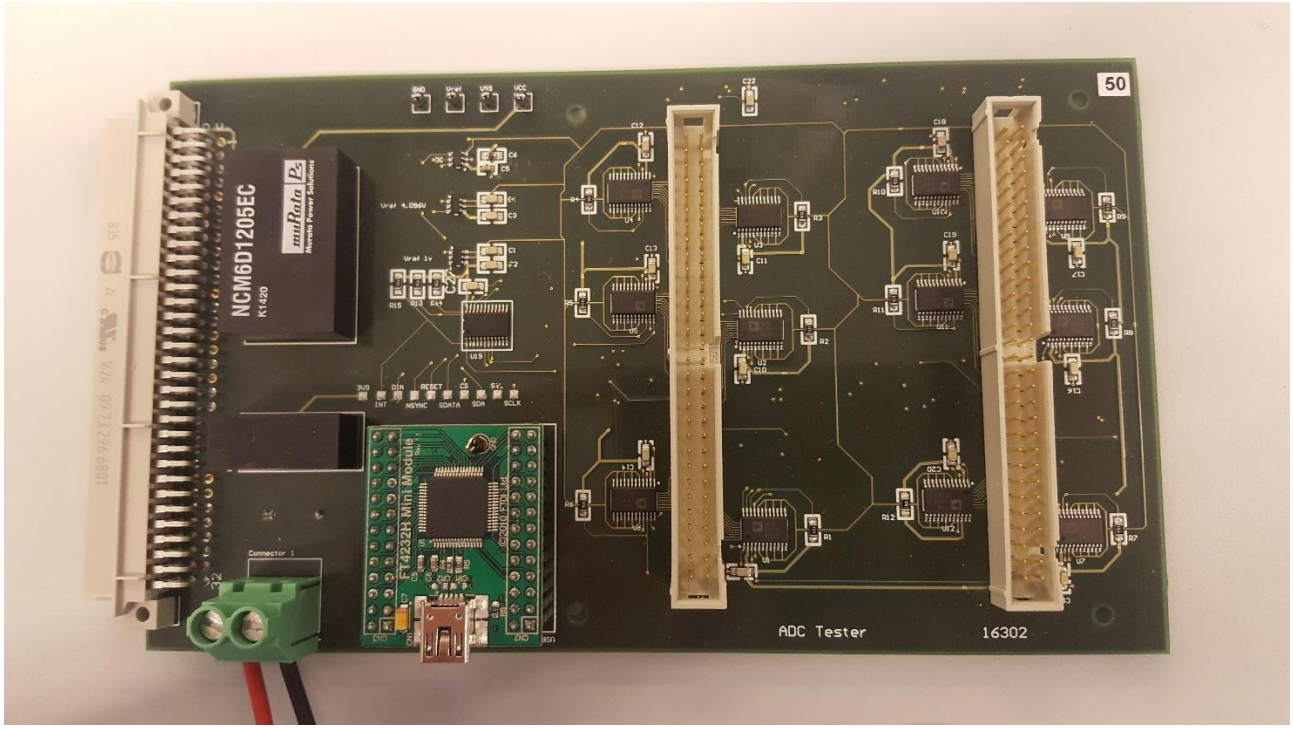
DESIGN OF A STAND ALONE ADC TESTER



Author:	Ewout Kesteloo
Organization:	HZ University of applied sciences
Student number:	047958
Email:	kest0005@hz.nl
Version:	1.0
Company:	NXP Semiconductors
In company mentor:	John van Zwam
Supervisory teacher:	Bert Verhage
Date:	09-01-17

DESIGN OF A STAND ALONE ADC TESTER

Author:	Ewout Kesteloo
Organization:	HZ University of applied sciences
Student number:	047958
Email:	kest0005@hz.nl
Version:	1.0
Company:	NXP Semiconductors
In company mentor:	John van Zwam
Supervisory teacher:	Bert Verhage
Date:	09-01-17

Summary

The Main question of the Thesis is: Which hardware and software is needed to test the ADC module of the EVA setup without the help of the EVA setup? Before answering the main question some background information is needed.

At NXP there is a bench test setup called the EVA. The EVA consist of several modules. If there is a malfunction in a board the board is normally replaced to be repaired at a later time. The module can be tested in the EVA setup but the work load is very high there is no time to test it in de setup. So the idea of an standalone tester was born.

For the design of the tester the SDM method is used. With this design method the assignment was split into smaller parts. By executing the different phases of SDM the tester was designed systematically.

The design is split in to two parts a hardware and software part.

The hardware part is designed with the following requirements

- Standalone (without the EVA set up)
- Self-test function
- Needs test pads to measure signals
- Connect to USB
- Connect to ADC module
- Wrong power connection protection a diode in the + of the connection

The design of the hardware is based on these requirements. The hardware is meant to be in a test environment.

The software is designed with the following requirements

- GUI needs to be easy in use
- The ADC tester needs to be able to test the 96 channels individually.
- Manual Testing
- Automatic testing

The software is designed that someone with limited knowledge can still operate the ADC tester.

With the hardware and software combined a working ADC Tester is created.

Foreword

The thesis project is the last part of the bachelor degree. For my thesis project I found a suitable assignment at NXP Nijmegen. The creation of a standalone ADC tester. The assignment was a combination of creating hardware and software. So all the disciplines that I learned at the HZ where used in the design of the ADC tester. During the thesis project John van Zwam was my in company mentor. Usually he had the answer to all my questions. I would like to thank him for his support. I also would thank Bert Verhage for the help and feedback.

This report shows how to design an ADC tester and starts with a single question: Which hardware and software is needed to test the ADC module of the EVA setup without the help of the EVA setup??

Abbreviations

- Analog to digital converter (ADC)
- Birds eye view (BEV)
- Chip select (CS)
- Clock (SCLK)
- Controller area Network (Can)
- Digital to analog converter (DAC)
- Device under testing (Dut)
- Data in (Din)
- General user interface (GUI)
- Inter-Integrated Circuit (I^2C)
- Local Interconnect Network (Lin)
- Most significant bit (MSB)
- Program structure diagram (PSD)Synchronization (NSYNC)
- System Development Methodology (SDM)
- Voltage reverence (Vref)

Table of Contents

Table of Figures

1. Introduction

The first part of the chapter gives information on the company where the thesis is conducted. The second part gives information on the thesis project.

1.1. Company Background

NXP® Semiconductors N.V. (NASDAQ: NXPI) enables secure connections and infrastructure for a smarter world, with advancing solutions that make lives easier, better and safer. As the world leader in secure connectivity solutions for embedded applications, NXP is driving innovation in the secure connected vehicle, end-to-end security & privacy and smart connected solutions markets. Built on more than 60 years of combined experience and expertise, the company has 45,000 employees in more than 35 countries.

NXP has design, research and development, manufacturing and sales operations in The Netherlands. NXP corporate offices are located at the Eindhoven High Tech Campus, a globally recognized hotspot of innovation. Eindhoven is home to core functions like Corporate Legal, Corporate Strategy, IT, Internal Communications, Sustainability Office, etc. The R&D activities in Eindhoven comprise the development of IP blocks, non-volatile memories, design tools & methodologies, test & verification methodologies, as well as product design in several Business Lines. R&D Strategy, Technology Scouting, External Relations and Standardization are functions based in Eindhoven. In Nijmegen there an research, development & business, manufacturing (8-inch wafer fab), technology enablement and support functions. (NXP, 2016)

1.2. Assignment

For bench testing, NXP is using an internal developed test setup called EVA. This is a common interface with on top (daughterboard) a dedicated product interface.

The whole setup is controlled by software written in Delphi, this is a windows based object oriented programming tool. With this tool, software can be written to measure the device types.

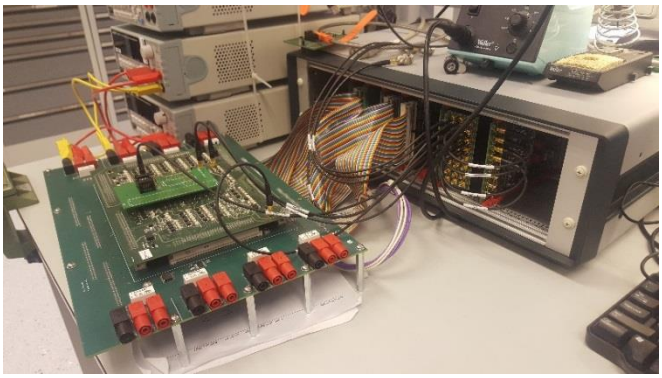


FIGURE 1 EVA TEST SETUP

The EVA setup (see figure one) exists of a backplane where several boards are plugged in. These boards e.g. DAC/ADC are used to control the DUT or measure parameters. If a board has a malfunction, then it needs to be changed by a new one. The board needs to be repaired, this cannot be done on de EVA setup because it is in use for production. To test the malfunction board a tool is needed to test it on the bench without the use of an EVA setup. The scope of the project maintains the ADC board of the EVA set-up.

The ADC board of the EVA setup is a plug in board that can easily be switched if its malfunctions.

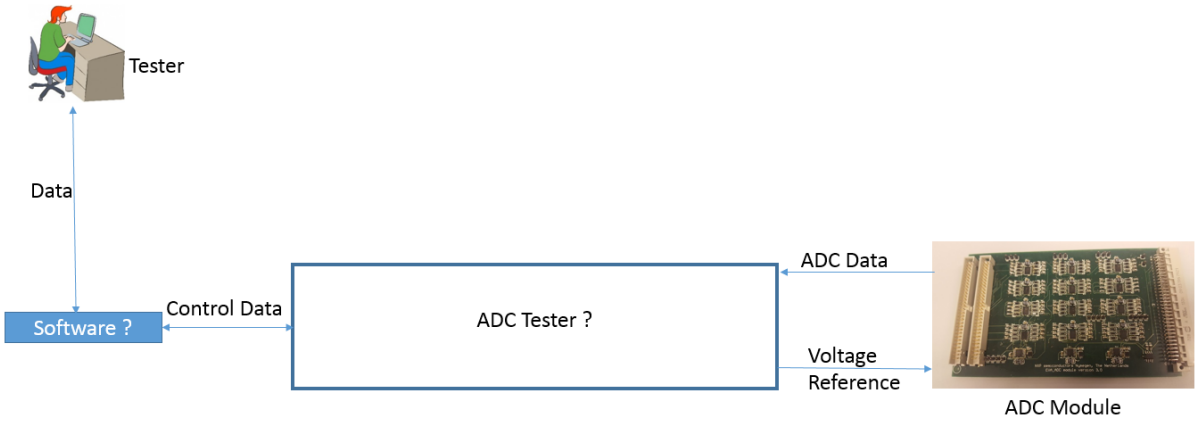


FIGURE 2 BIRDS EYE VIEW (BEV)

In figure two the Birds eye view of the assignment is shown. There are two components with a question mark these are the components that need to be designed.

The software that controls the tester and gives read outs to the tester. The hardware that makes the physical connections and controls all the signals. The voltage reference in the BEV is there to get a reliable voltage source to the ADC module.

1.3. Questions

The main question is: Which hardware and software is needed to test the ADC module of the EVA setup without the help of the EVA setup?

In order to be able to answer the main question it is split into multiple sub questions.

The sub questions are determined according to the SDM design method. First the problem is determined in the problem confrontation phase. In the problem analyst phase the problem is analyzed. Subsequently a design is made to solve the problem in the functional design phase. In the physical design phase the design is executed. The SDM method is explained more elaborately in chapter 3.

The problem confrontation phase

- What is an ADC?
- Which ADC is on the EVA plugin board?
- What is fan out?
- What is bit banging?

The problem analyst phase

- How does the ADC on the Eva board work?
- What for components are used on the ADC boards?
- On what voltage does the ADC operate?
- What needs to be measured?

The functional design phase / the definition phase

- How to control the ADC
- How to control the tester?
- What signals will the tester use?

Physical design phase

- How to connect the ADC module to the ADC tester

The sub questions of the problem confrontation phase are answered in the theoretical framework in the next chapter.

2. Theoretical Framework

This theoretical framework contains a brief overview of the theory.

2.1. Software

The following software is used during the thesis project.

- Delphi 10, which is a windows based object oriented programming tool – this is used to write all the software for the project. The software is written in Pascal.
- Altium designer is used for the design of the hardware.
- Microsoft office 365 is used to create all the documentation and presentations.

2.2. ADC / DAC

An analog-to-digital converter, or ADC, converts analog (continuously variable) signals into digital (multi-level) signals without altering their essential content. The analog information is transmitted by modulating a continuous transmission signal and by amplifying the strength of the signal or varying its frequency in order to add or subtract data. The input to an ADC (analog) consists of a voltage which varies in a range of theoretically infinite number of values. The output of the ADC (digital) however has defined levels or states. Digital signals are transmitted in a more efficient way than analog signals because well-defined digital impulses are easier to distinguish for an electronic circuit than chaotic noise. This is the main advantage of digital communications. (Malik, 2016)

Digital to analog converting is a process where digital signals that have a few (usually two) defined states are turned into analog signals, which have a theoretically infinite number of states. A Digital to Analog Converter, or DAC, is an electronic device that converts a digital code to an analog signal such as a voltage, current, or electric charge. Signals can easily be stored and transmitted in digital form; a DAC is used for the signal to be recognized by human senses or non-digital systems. Converting a signal from digital to analog can degrade the signal. Therefore details are chosen so that errors are negligible. Due to their cost, digital to analog converters are mostly manufactured on an integrated circuit (IC). DAC architectures may contain different advantages as well as disadvantages. The suitability of a digital to analog converter for a particular application is determined by several attributes such as speed and resolution. (digital-analog-conversion, 2016)

The DAC on the DAC board of the EVA setup is the AD5328. The AD5328is a 12-bit buffered voltage output DAC in a 16-lead TSSOP. The AD5328 uses a versatile 3-wire serial interface that operates at clock rates up to 30 MHz and is compatible with standard SPI, QSPI, MICROWIRE, and DSP interface standards. (DAC, 2016)

The ADC on the ADC board of the EVA setup is the AD7927. The AD7927 is a 12-bit, high speed, low power, 8-channel, successive approximation ADC. The conversion process and data acquisition are controlled using CS and the serial clock signal, allowing the device to easily interface with microprocessors or DSPs. The input signal is sampled on the falling edge of CS and the conversion is also initiated at this point. (ADC, 2016)

2.3. Successive approximation ADC

The successive approximation converter shown in Figure 3 operates by approximating the analogue input signal with a binary code. Binary codes are the most simple digital signals with two states. This binary code is successively revising by changing each bit in the code until the best approximation is achieved. At each step in the approximation, the present estimate of the binary value corresponding to the analogue input signal is saved in the successive approximation register. The contents of this

register are converted to an analogue signal by a DAC so that a single comparator can determine whether the approximation is larger or smaller than the input signal. As shown in Figure 3 the first approximation sets the most significant bit, the MSB, of the successive approximation register and resets all the other bits (i.e. makes them zero). If the DAC output (which is therefore equal, at this point, to half full-scale) is smaller than the analogue input, the MSB is left on; if the DAC output is too large, then the MSB is turned off. In the next clock cycle, the next most significant bit is set (i.e. at the DAC output is now equal to either 3/4 or 1/4 of full-scale, depending on whether the most significant bit was left on or not) and this new approximation is compared with the analogue input. Each successive bit is similarly tested. After the least significant bit has been tested, the conversion is complete and the output register contains the binary code. (robots.ox.ac.uk, 2016)

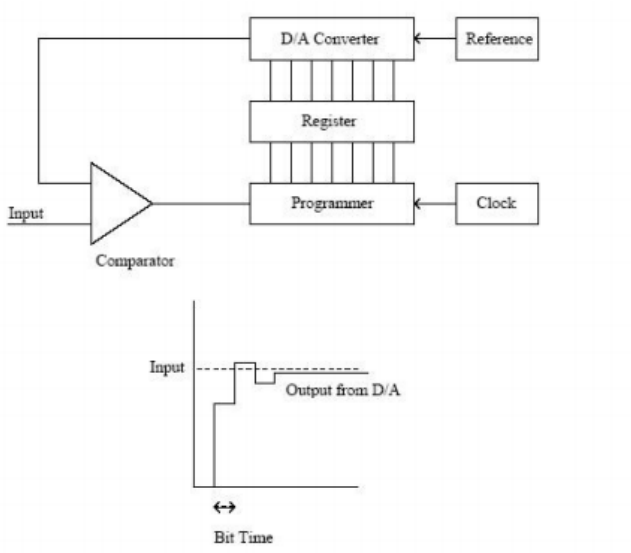


FIGURE 3 A SCHEMATIC DIAGRAM OF THE ARCHITECTURE OF SUCCESSIVE APPROXIMATION

2.4. I²C

The physical I²C bus consists of two wires, called SCL and SDA. SCL is the clock line. It is used to synchronize all data transfers over the I²C bus. SDA is the data line. The SCL and SDA lines are connected to all devices on the I²C bus. There needs to be a third wire which is the ground or 0 Volts. There may also be a 5 Volt wire for power distribution to the devices. Both SCL and SDA lines are "open drain" drivers. Therefore the chip can drive its output low, but it cannot drive it high. For the line to be able to go high, pull-up resistors to the 5V supply must be provided. For the complete I²C bus one set of pull-up resistors is needed. This is due to the fact there needs to be a resistor between the connection of the 5V with the SCL and the 5V with the SDA and not for each device. This is illustrated in the figure below.

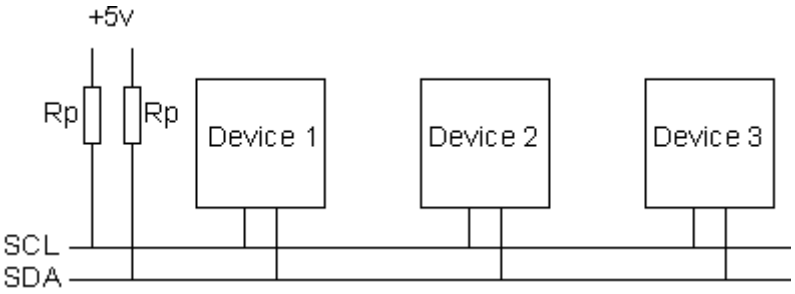


FIGURE 4 I²C BUS

The devices on the I²C bus are either masters or slaves. A master drives the SCL clock line, the slaves respond to the master. A master can initiate a transfer over the I²C bus, a slave cannot. Usually, there is one master on a I²C bus with multiple slaves. (i2c-tutorial, 2016)

2.5. Bit banging

Bit banging is a technique for serial communications using software instead of dedicated hardware. Software directly sets and samples the state of pins on the microcontroller, and is responsible for all parameters of the signal: timing, levels, synchronization, etc. In contrast to bit banging, dedicated hardware (such as a modem, UART, or shift register) handles these parameters and provides a (buffered) data interface in other systems, so software is not required to perform signal demodulation. Bit banging can be implemented at very low cost. (Bit banging, 2016)

2.6. Fan Out

Fan Out is defined as the maximum number of inputs (load) that can be connected to the output of a gate without degrading the normal operation. Fan Out is calculated from the amount of current available in the output of a gate and the amount of current needed in each input of the connecting gate. It is specified by manufacturer and is provided in the data sheet. Exceeding the specified maximum load may cause a malfunction since the circuit will not be able supply the demanded power. (difference-between-fan-in-and-fan-out-in-digital-electronics, 2016)

2.7. Software design

The software design in this theses will be executed with Nassi–Shneiderman diagrams. A Nassi-Shneiderman diagram is used for representing the sequence of execution in a program. The diagram takes the form of a rectangle divided mainly into smaller rectangles with the sequence of execution going from top to bottom of the diagram. There are various standard constructs, including NS sequence structures, NS repetition structures, and NS selection structures. (A Dictionary of Computing, 2017). These structures are expressed by three basic symbols:

- Process(Sequence)
- Decision(Selection)
- Iteration(Repetition)

Process

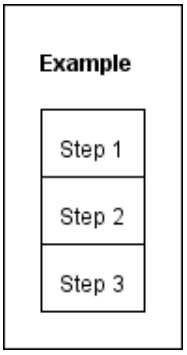


FIGURE 5 EXAMPLE PROCESS

In figure 5, a simple process is shown. The three boxes represent three steps the program is going to execute.

Decision

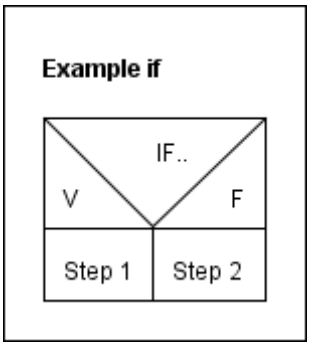


FIGURE 6 EXAMPLE IF PROGRAM

In figure six the if statement is shown. When the if the statement is true step one will be executed. When the statement is false step 2 will be executed.

Iteration

In figure 7 and 8 the loop forms of the Nassi-Shneiderman diagrams are shown. There are two types of loop forms: a while loop (Figure 7) and a repeat loop (Figure 8). A while loop checks the conditions before executing the loop and the repeat loop checks the conditions after the loop is executed.

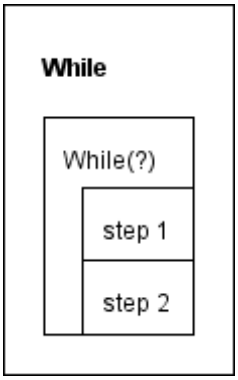


FIGURE 7 WHILE LOOP

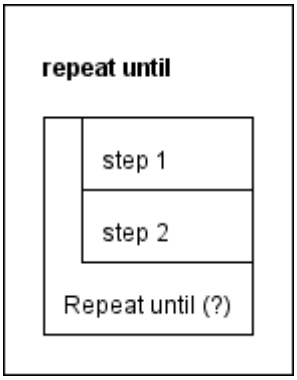
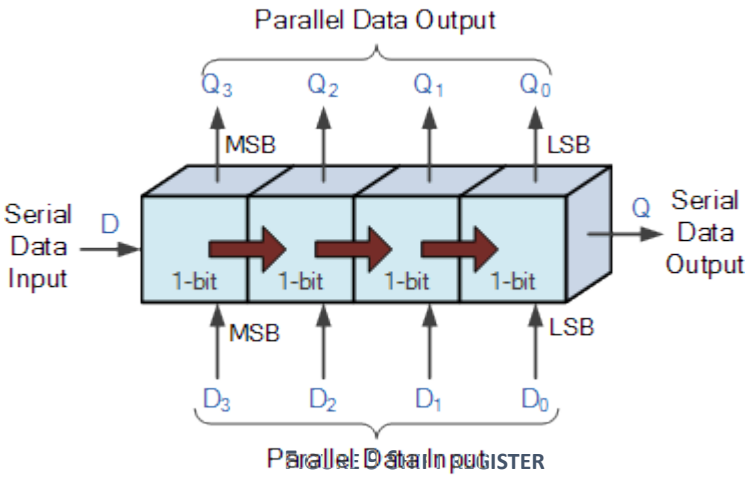


FIGURE 8 REPEAT UNTIL LOOP

2.8. Shift register

A Shift Register is another type of sequential logic circuit that can be used for storage or transfer of data in the form of binary numbers.

This sequential device loads the data present on its inputs and then moves or "shifts" it to its output once every clock cycle, hence the name Shift Register.



A shift register basically consists of several single bit "D-Type Data Latches", one for each data bit, either a logic "0" or a "1", connected together in a serial type daisy-chain arrangement so that the output from one data latch becomes the input of the next latch and so on. (electronics-tutorials, 2017)

3. Method

The research documentation is structured according to the System Development Methodology (SDM) method.

The System Development Methodology was created by the company PANDATA in 1970. Commissioned by three big companies in the Netherlands (Akzo, Nationale Nederlanden and former PTT). SDM is a methodology for planning, designing, building, implementing and managing information systems.

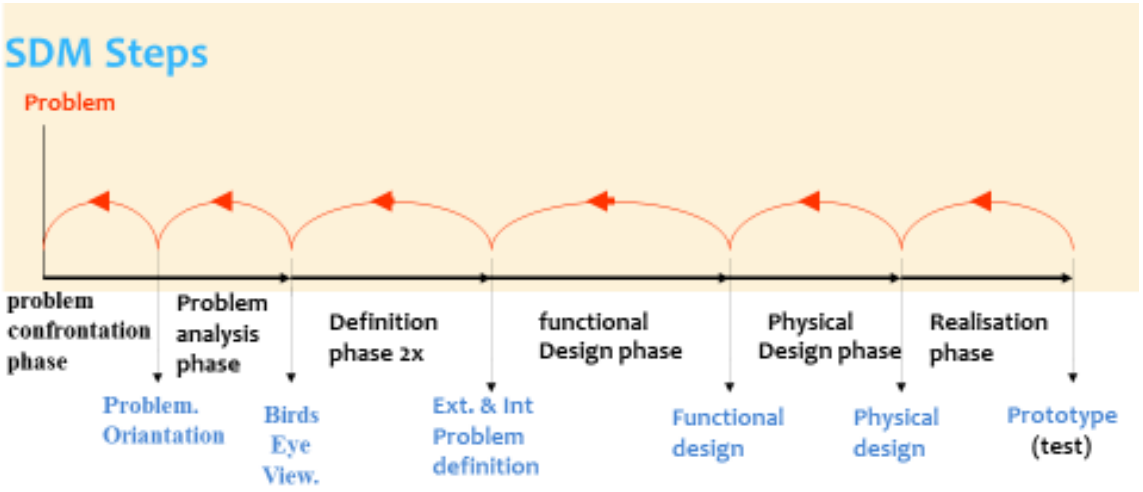


FIGURE 10 SDM STEPS

SDM consist of several steps that need to be taken in order to achieve a prototype. These steps are visually represented by the black lines in figure 10. If there is a complication during the execution of the phase, there is always the possibility to go back and change something in the phase before (iterate). This is represented by the red lines in Figure 10.

The steps that can be recognized are:

- Problem confrontation phase
- Problem analysis phase
- Definition phase
    - External problem definition
    - Internal problem definition
- Functional design phase
- Physical design phase
- Realization phase

Every phase leads to intermediate products.

- The problem confrontation phase → Problem orientation
- The problem analysis → Birds eye view
- External problem definition → External problem definition
- Internal problem definition → Internal problem definition
- Functional design phase → Functional design
- Physical design phase → Physical design
- Realization phase → Prototype

Enumeration of the interim products

- Problem orientation
  - a clear description of the problem
- Birds eye view
  - A visual representation of the problem. The bird's eye view will make the problem more clear. It can also be used as an intermediary when discussing the problem.
- External problem definition
  - All inputs defined
  - All outputs defined
  - A functional title of all components to design
- Internal problem definition
  - All parts defined
  - Internal structure visible
  - All components divided into simple problems
  - All internal connections defined
- Functional design
  - Structure visible of the solution
  - Structure matches the internal problem definition
  - User friendly
  - Operation friendly
  - Safety
- Physical design
  - All sub functions are technical specified
  - A principle schematic
  - A circuit board design
  - Design documents
- Prototype
  - A prototype

3.1. Justification

SDM is chosen for the fact that it is user friendly with clear steps which need to be taken during a design process. First the birds eye view results in a visual representation of the problem. Hereafter the problem is divided into smaller problems by defining all connections in the external problem definition. These smaller problems can be solved more easily. This is realized by zooming in on the problems and subsequently creating an internal problem definition. From the internal problem definition the functional design is created.

This project is a combination of hardware and software. SDM is originally created to design software, though it is also applicable for the design of the ADC. This is because all the connections are defined in the external and internal problem definition.

4. Results

In this chapter the result of the thesis project will be described in the order of the SDM method. The sub questions are answered in every phase of the method.

4.1.1. End product definition

As an end product, NXP would like a standalone tester for the ADC module board of the EVA setup. In order to make the design meets the expectations of NXP, the expectations were discussed during a meeting with the company mentor. These expectations were set as requirements for the design. The requirements are listed here.

- Standalone (without the EVA set up)
- Able to measure signals with test pads
- Able to connect to USB
- Able to connect to ADC module
- Wrong connection protection (a diode in the + of the connection)
- User friendly GUI
- The ADC tester is able to individually test the 96 channels
- Possibility of manual testing
- Possibility of automatic testing

This list of requirements can be split in two parts a hardware and a software part.

The hardware requirements are:

- Standalone (without the EVA set up)
- In possession of a self-test function
- Able to measure signals with test pads
- Able to connect to USB
- Able to connect to ADC module
- Wrong connection protection (a diode in the + of the connection)

These requirements are taken into account while selecting components for the ADC tester.

The software requirements are:

- User friendly GUI
- The ADC tester is able to individually test the 96 channels
- Possibility of manual testing
- Possibility of automatic testing

The definition of a user friendly GUI Is that the GUI needs to be clear and self-explanatory so the tester can begin testing without having to read a manual.

4.2. External overview

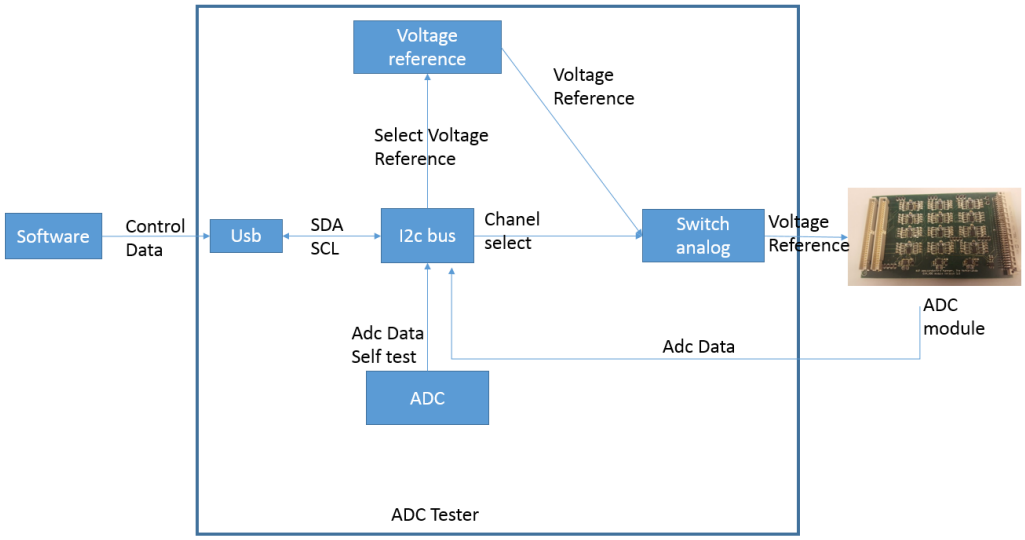


FIGURE 11 ADC TESTER EXTERNAL OVERVIEW HARDWARE

In the external overview the components which need to be selected are visualized. Each component has its own requirements. The external overview of the hardware is depicted in Figure 11. The USB module connects the ADC tester to the computer via an USB connection. To control the tester all data will be send by I²C. Therefore, the USB module needs to support the I²C protocol. To be able to test if the ADC module is functioning correctly a voltage reference is needed in order to obtain a constant input to the ADC module. The ADC needs to have a high output impedance allowing placing of two different voltage references on the ADC tester. Therefore the voltage reference is the heart of the ADC tester. To test all the pins individually there is an analog switch needed to switch the 96 pins to the voltage reverence. To be able to tell if the ADC tester still functions an ADC is placed on the tester. With this ADC the voltage reference can be checked.

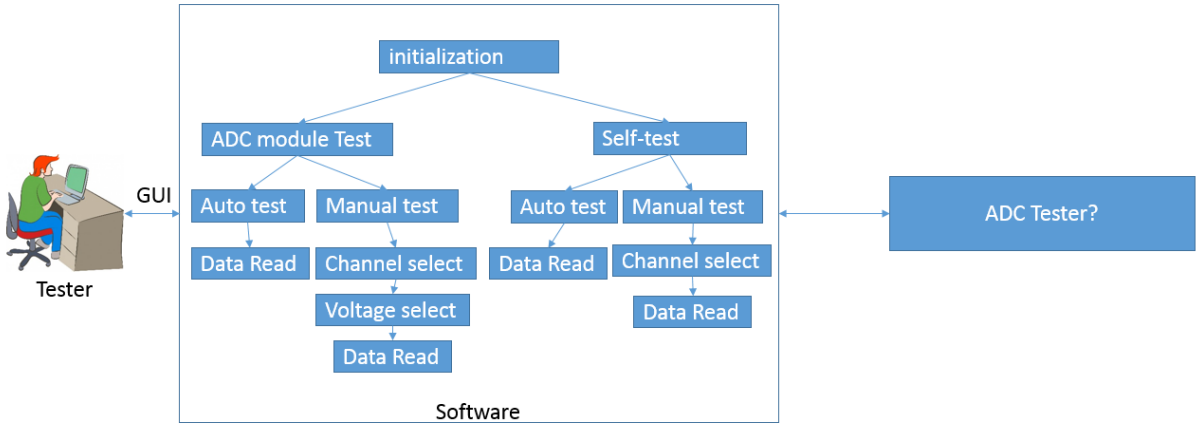


FIGURE 12 EXTERNAL OVERVIEW SOFTWARE

Figure 12 shows the external overview of the software. Here two paths are shown. A path to test the ADC module and a path to test the ADC tester itself. Both of the programs need an auto test function to automatically check the functions. Both also contain a manual function to test the channels individually.

4.3. Internal overview

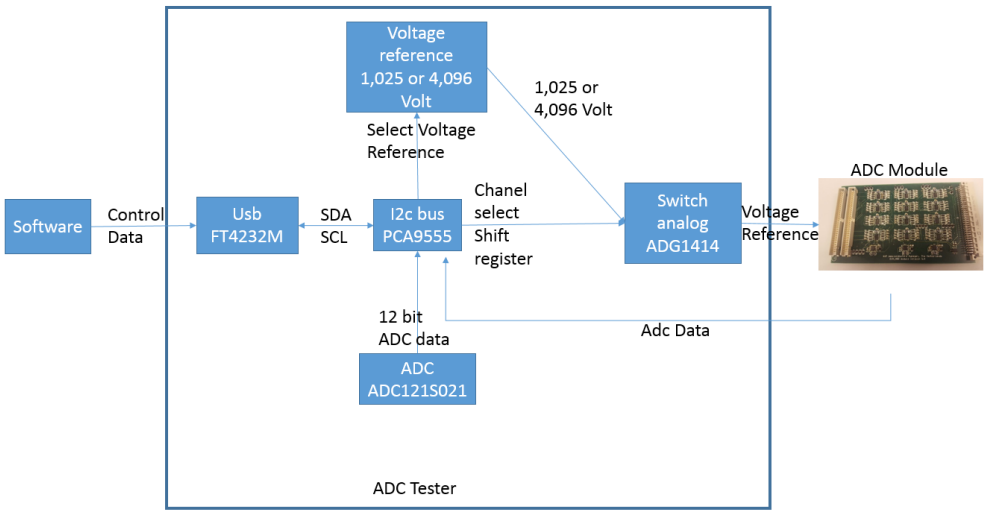


FIGURE 13 ADC TESTER INTERNAL OVERVIEW HARDWARE

The internal overview is the last step before the actual design of the hardware. In the internal overview shown in figure 13 all the components and connections are known and defined. The USB module is there for the connection to the pc and is the master of the I²C bus. With the I²C bus all the components are controlled. The switches are present to connect the voltage reference to the correct channel of the ADC module. Hence there are 96 analog switches on the ADC tester. The switches are selected with a 96 bit shift register controlled by the I²C bus. To power all the components the DC-DC converter is placed. The switches are operating on -5V and +5 Volt so a DC-DC converter for this voltage was selected.

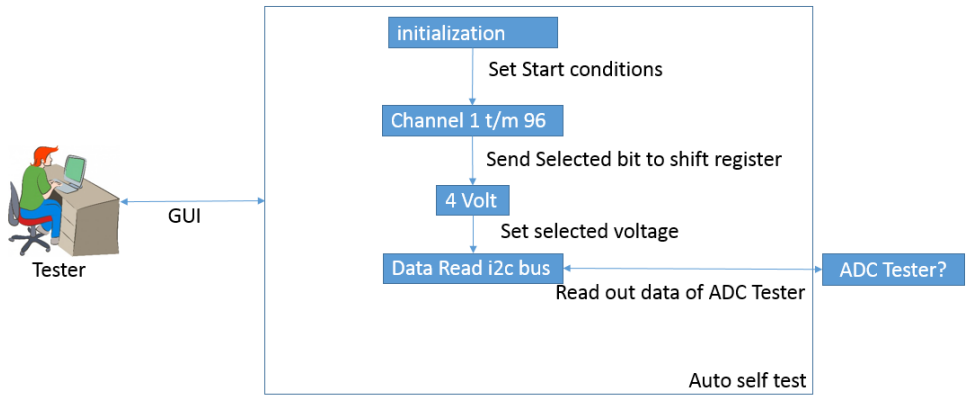


FIGURE 14 INTERNAL OVERVIEW AUTO SELF-TEST

In figure 14 the internal overview of the automatic self-test can be seen. The automatic self-test function is one function of the ADC tester software. During the automatic self-test the 96 channels are tested individually. This is executed by selecting the switch and then applying 4 Volt to it and test it via the I²c bus. If the I²c bus gives a "1" the switch is working correctly and indicates an ok if the switch fails it gives a "0" and indicates there is a fail.

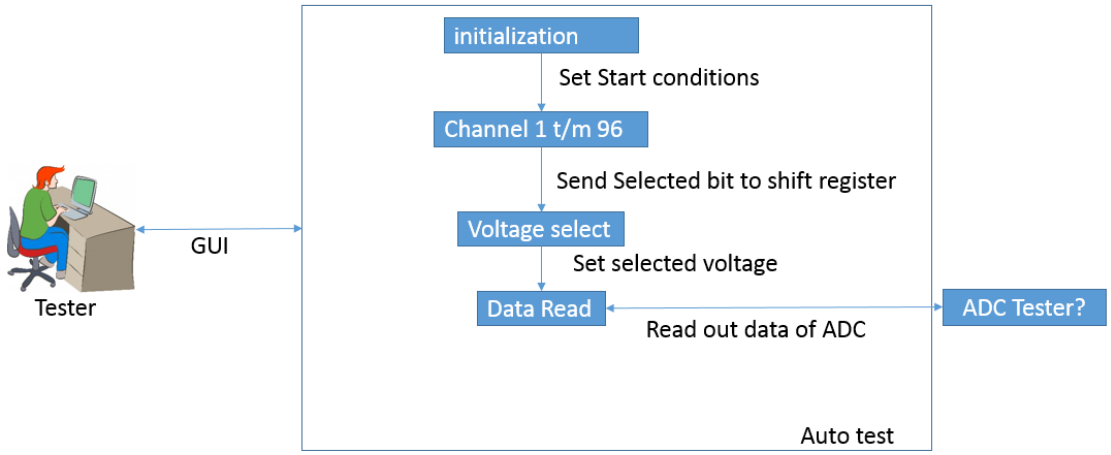


FIGURE 15 INTERNAL OVERVIEW AUTO TEST

In figure 15 the internal overview of the automatic test can be seen. The automatic test function is one of the main functions of the ADC tester software. During the automatic test the 96 channels are tested individually. By selecting the switch and then testing it on two voltage levels, namely 4.096 Volt and 1.025 Volt. If both voltage levels of the check are correct the software gives out an ok otherwise it gives a fail.

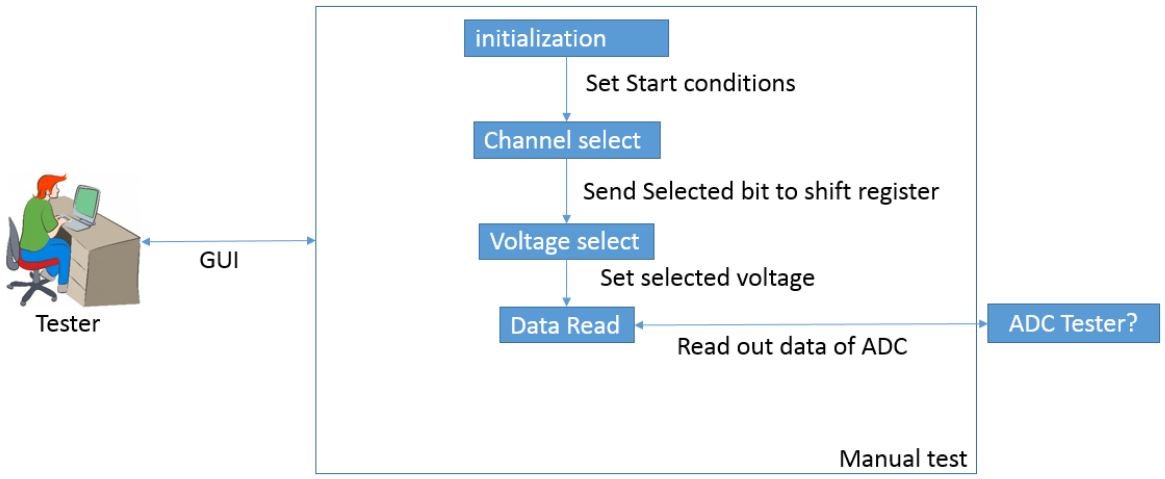


FIGURE 16 INTERNAL OVERVIEW MANUAL TEST

In figure 16 the internal overview of the manual test is shown. The manual test displays the different procedures the software needs to execute. With the manual test, the channel and voltage can be selected for testing.

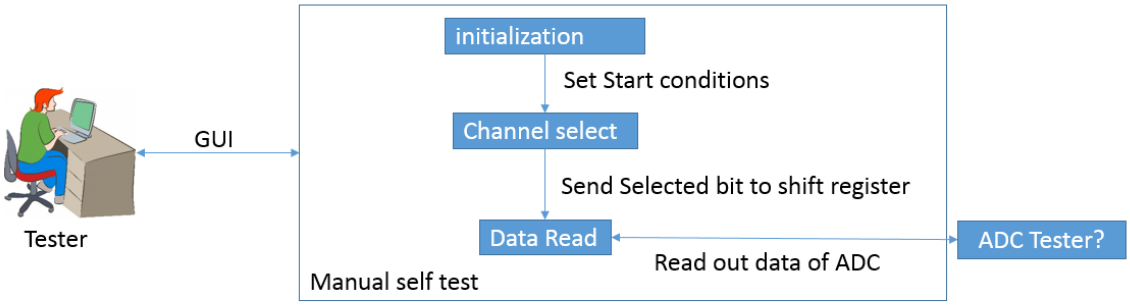


FIGURE 17 INTERNAL OVERVIEW MANUAL SELF-TEST

In figure 17 the internal overview of the manual test is shown. The manual test displays the different procedures the software needs to execute. The manual self-test is nearly the same as the automatic self-test with the difference that the tester selects the channel to be tested.

4.4. Functional design

In this chapter the design choices are shown. In the first part of this chapter the hardware decisions will be explained. In the second part of the chapter the software decisions will be explained.

4.4.1. Hardware

The hardware design consists of multiple components which together comprise the complete ADC tester. The hardware is logically constructed. The first components to be elected were the analog switches in order to be able to select the inputs of the ADC module one by one.

Analog switch

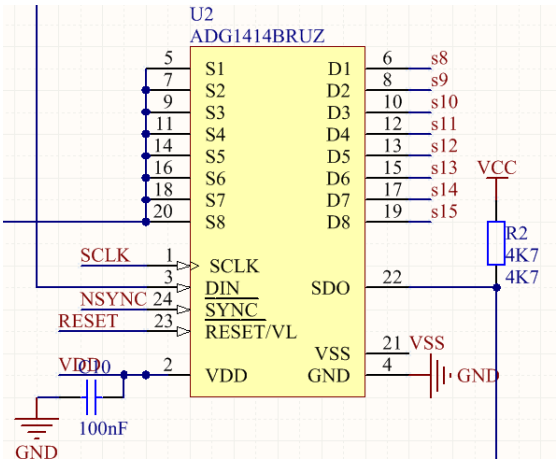


FIGURE 18 ANALOG SWITCH

After some research the ADG 1414 was chosen because this is an analog switch with the capabilities to daisy chain all the switches together so only one command needs to be sent to change the state of one of the switches. The switches are analog because the voltage of the voltage reference needs to be carried to the ADC module and not a “1” or a “0”. One side of the switches is connected to the voltage reference and the other side leads to the ADC module see figure 18. There is a switch present for every channel, so each channel can be tested individually. All the switches are daisy chained together. This means that the data which enters the first switch is passed on to the next switch until all 96 bits are send. The switches can run on three different voltages. +15V -15V, +12V and -5V +5V. The best option

for the ADC tester is -5V +5V because the +5V can be used by all the other components as well. The SCLK is the clock input of the analog switch. It can be programmed together with the N-sync and the Din.

DC-DC converter

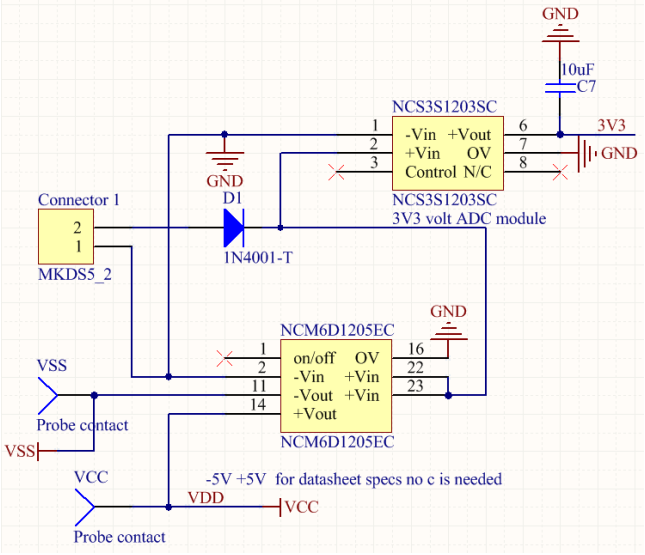


FIGURE 19 DC-DC CONVERTERS

The ADC tester contains two DC-DC converters. The two DC-DC converters deliver three different voltages, namely +5V, -5V and 3,3 Volt. The DC-DC converters are selected based on the specifications needed for the design. The ability to deliver 3,3 Volt is needed on the ADC module because some of the components on the module run on 3,3 Volt. The -5V and +5 Volt are especially required for the switches. The +5V can be used by all the other components as well. The voltages are defined in the schematic seen in figure 19 as follows

- +5 Volt = VDD / VCC
- -5 Volt = VSS
- 3,3 Volt = 3V3

Voltage Reference

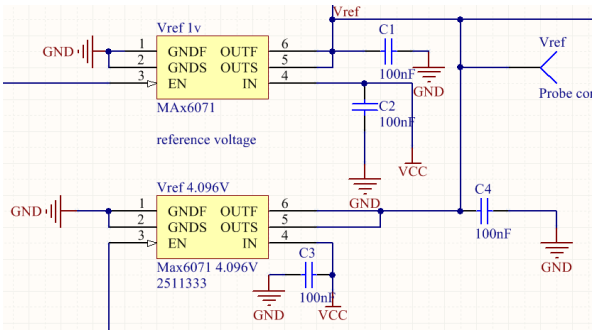


FIGURE 20 VOLTAGE REFERENCE

The voltage reference is the most important object because all the measurements depend on the voltage to be correct. As voltage reference, the max 6071 is chosen for this design for the reason that

this voltage reference has a high impedance output when not in use. Therefore two voltage references can be present in the scheme.

The voltage reference contains an EN pin which stands for enable pin see figure 20. This pin is connected to the I²C bus expander in order to be able to switch the voltage reference. This is necessary to have the possibility to test the ADC on two voltage levels. The voltage levels which were chosen were 1.025 Volt and 4.096 Volt. With these levels a low and a high point can be measured.

I²C bus expander

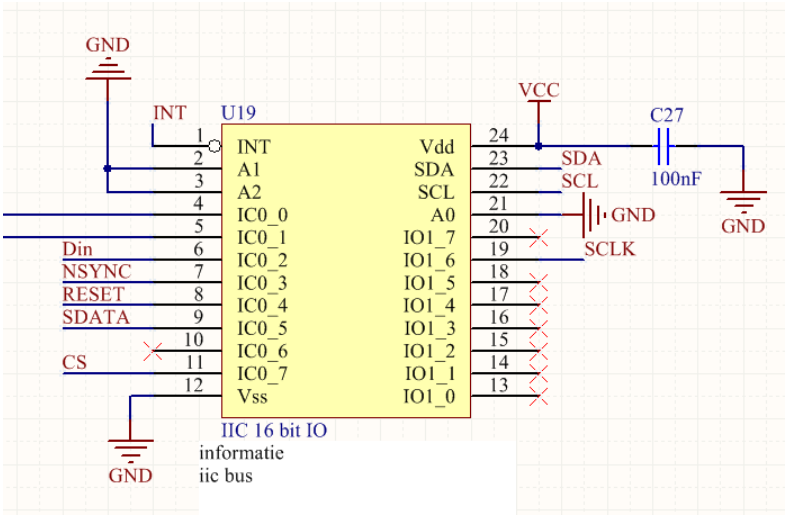


FIGURE 21 I²C BUS EXPANDER DATA

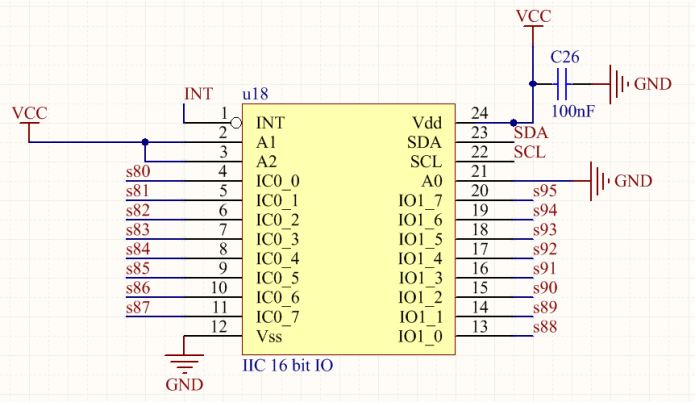


FIGURE 22 I²C BUS EXPANDER SWITCH+

The IO expander also mentions as the I²C bus. Are an input/output expander chip that can be controlled via I²C. There are 7 IO bus expanders on the ADC tester. All the IO expanders have their own address to be able to communicate with the expanders individually via the I²C bus.

The first bus expander is needed for controlling the data of the ADC tester. The data is send out via bit banging so the IO expander can control the other components. The other six IO expanders are necessary for the self-test function of the ADC tester. The IO expander can register a “1” or a “0” via the input function. The voltage reference of 4,096 Volt is sufficient to input a “1”. Therefore with this function the 96 switches can be tested.

USB Mini Module

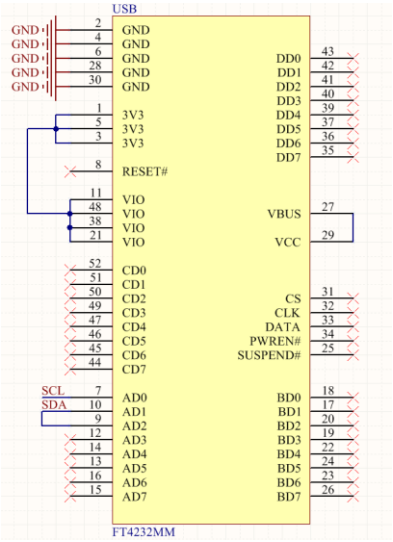


FIGURE 23 USB MINI MODULE

To transfer the data to the computer the FT4232 mini module is chosen see figure 23. This device was available since it is used in several equipment designed by NXP and it was suitable for this design as well. The entire communication to the computer of the tester will be handled with the FT4232 mini module. The FT4232 mini module is equipped to use I²C and has a specific I²C output which gives it the ability to connect to IO expanders.

ADC

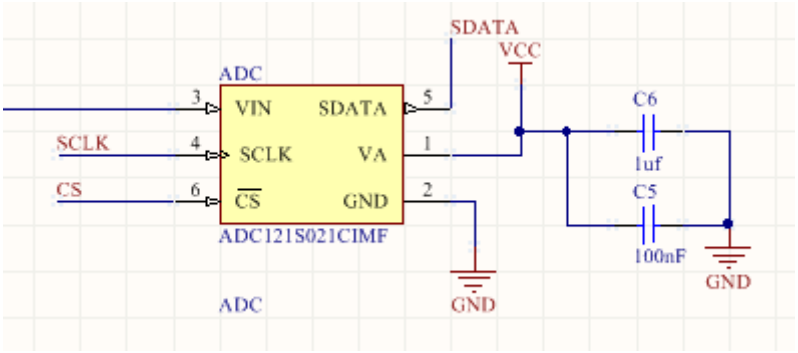


FIGURE 24 ADC

To have the possibility to test if the voltage reference is working correctly, the ADC tester includes an ADC see figure 24. The ADC chosen on the ADC tester is an ADC121S021CIMF which is a low power single channel cmos 12-bit analog to digital converter. This ADC is chosen because of the same bit rate as the ADC`s present on the module. The ADC also operates on +5 Volt.

For a full image of the scheme, all the objects are combined in appendices one and two

4.4.2. Software

The design of the software is divided in separate functions and procedures. This gives the advantage that the program is easy to read and when an adjustment is needed, it will only need to be adjusted once.

The functions of the design are:

- Initialization
- Clock signal
- Shift register fill
- ADC read ADC tester
- Control Voltage Reference
- Auto Self-test
- Manuel Self-test
- Auto test ADC module
- Manuel Test ADC module

With these functions the software can be programmed.

Initialization

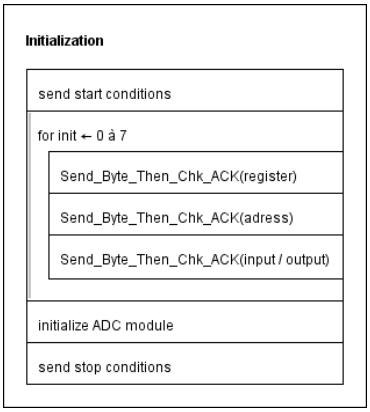


FIGURE 25 INITIALIZATION

TABLE 1 ADDRESSES OF THE DIFFERENT FUNCTIONS OF THE IO BUS EXPANDER.

Pin	Data	Address	Register	Status
Ic0-0	Vref 1Volt	0000 0001$1	0	Output
Ic0-1	Vref 4 Volt	0000 0010 $2	0	Output
Ic0-2	Din	0000 0100 $4	0	Output
Ic0-3	NSYNC	0000 1000 $8	0	Output
Ic0-4	RESET	0001 0000 $10	0	Output
Ic0-5	SDATA	0010 0000 $20	0	Input
Ic0-7	CS	100 00000 $80	0	Output
IO1_6	SCLK	0100 0000 $40	1	Output

The first procedure the software needs to execute is the initialization of the components see figure 25 for the PSD. The I²C modules are normally all outputs. These need to be set to inputs before the software can read out the data. The first IO expander is used to control the data for the ADC tester, see table 1 for the connections. On the first IO expander the in- and outputs need different settings than the other IO expanders. To set the pins on output the correct data need to be send. First the

address of the IO expander needs to be send. Then the right register needs to be selected. After that the settings can be send to the IO expander.

Clock signal

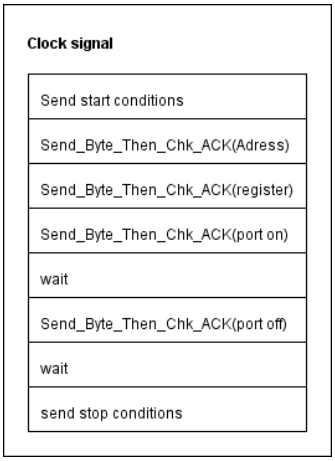


FIGURE 26 PSD CLOCK SIGNAL

The clock pulse function is present in order to be able to control the components via bit banging. To create a clock signal the pin connected to the clock needs to be toggled. The clock is connected to pin 6 of the second register of the first I²C expander. The wait function in the PSD figure 26 is present to make sure the other components notice the clock signal. Additionally the clock signal needs to be combined with the data otherwise the clock pulse will not be noticed and a zero will be detected.

Auto Self Test

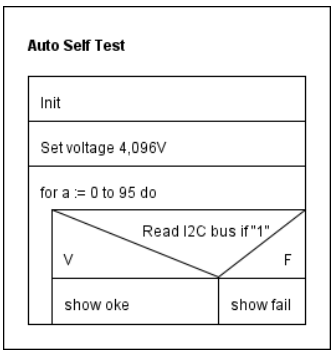


FIGURE 27 PSD AUTO SELF-TEST

As part of the requirements the ADC tester needs to be able to test itself. The hardware on the ADC tester is chosen to fulfill this requirement. The six I²C expanders are present to read out the data from the switches. If the voltage reference is set on 4 Volt the digital input of the I²C expander reads it as a "1". With this data the ADC tester can be tested. For a simple PSD of the auto self-test function see figure 27

Control voltage reference

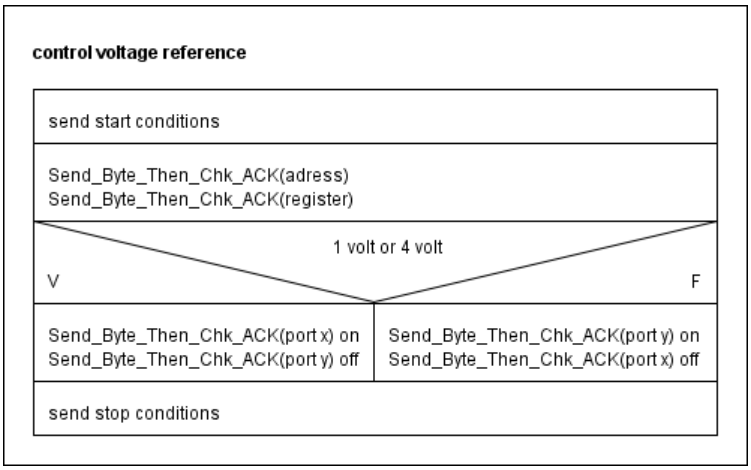


FIGURE 28 PSD CONTROL VOLTAGE REFERENCE

The voltage references are connected to pin 0 and pin 1.

To be guaranteed that only one voltage reference is active on the ADC tester. It needs to be impossible to set both references on at the same time. Therefore, the ports need to switch at the same time. The pin of the selected voltage reference needs to be high at the time of the measurement. For a simple PSD of the procedure see figure 28

Auto Test ADC module

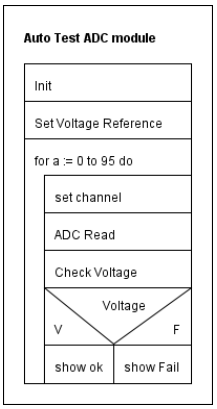


FIGURE 29 AUTO TEST ADC MODULE

In figure 29, the PSD of the automatic test function is shown. The auto test function needs to complete several steps to be able to test. The automatic test needs to test the 96 channels individually. There for every channel needs to be set in order. The next step is to read the ADC module and check if the read out is corresponding to the set voltage reference.

Manual Self-test

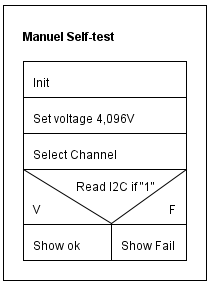


FIGURE 30 PSD MANUAL SELF-TEST

The manual self-test function of the software is almost the same as the automatic program. The only difference is that the voltage and channel can be selected. For a simple PSD of the manual test function see figure 30

Shift register fill

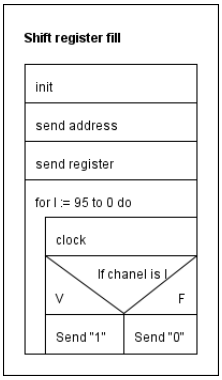


FIGURE 31 PSD SHIFT REGISTER FILL

The 96 switches are controlled via a shift register. The first bit in the shift register is corresponding to switch 96. The last bit send corresponds with the first switch. With this knowledge the software procedure can be created to be able to switch a specific switch on or off. For a simple PSD of the shift register fill procedure see figure 31.

ADC Read

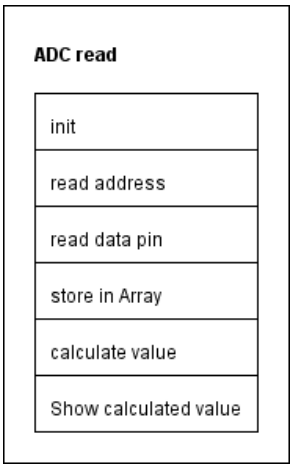


FIGURE 32 PSD ADC READ

The ADC on the tester is a 12-bit single channel ADC. To read out the ADC via the IO expander the software needs to read 12 bits from the ADC. These 12 bits need to be stored in an array. With the data the voltage can be calculated. Then the voltage can be displayed. The steps the procedure needs to execute are in the PSD shown in figure 32.

Manual Test

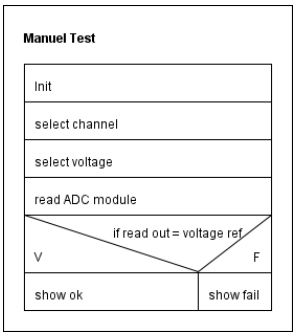


FIGURE 33 PSD MANUAL TEST

The manual test of the ADC module the program is very similar to the automatic test PSD. The main difference is that in the manual test the channel and voltage are manual selected. Figure 33 shows a simple PSD of the manual test function.

4.5. Physical design

In the physical design phase the final design is created. The physical design is split in two parts, namely the software part and the hardware part.

4.5.1. The Software

The final design of the software part.

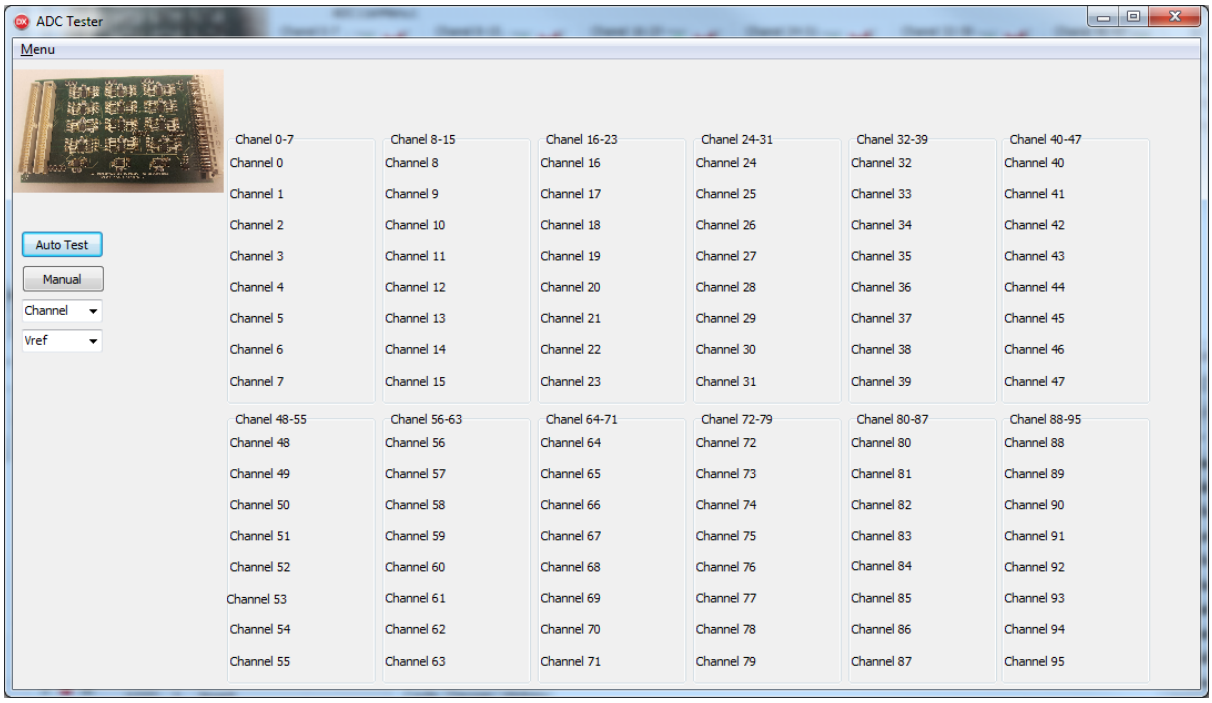


FIGURE 34 GUI MAIN

The GUI is the visual aspect of the program see figure 34. With this the tester will interact with the ADC tester hardware. There are two buttons on the screen which represent an auto test and a manual test. With the auto test button the ADC module will be tested automatically on 1,025 Volt and 4,096 Volt. When both voltages are measured the program gives an ok as shown in figure 35. If the ADC does not give the correct voltages the program gives a fail as shown in figure 36.

FIGURE 35 OK

FIGURE 36 FAIL

With the manual test function of the program the channel can be selected. In addition, the voltage of the voltage reference can be selected which gives the ability to test the ADC on two levels.

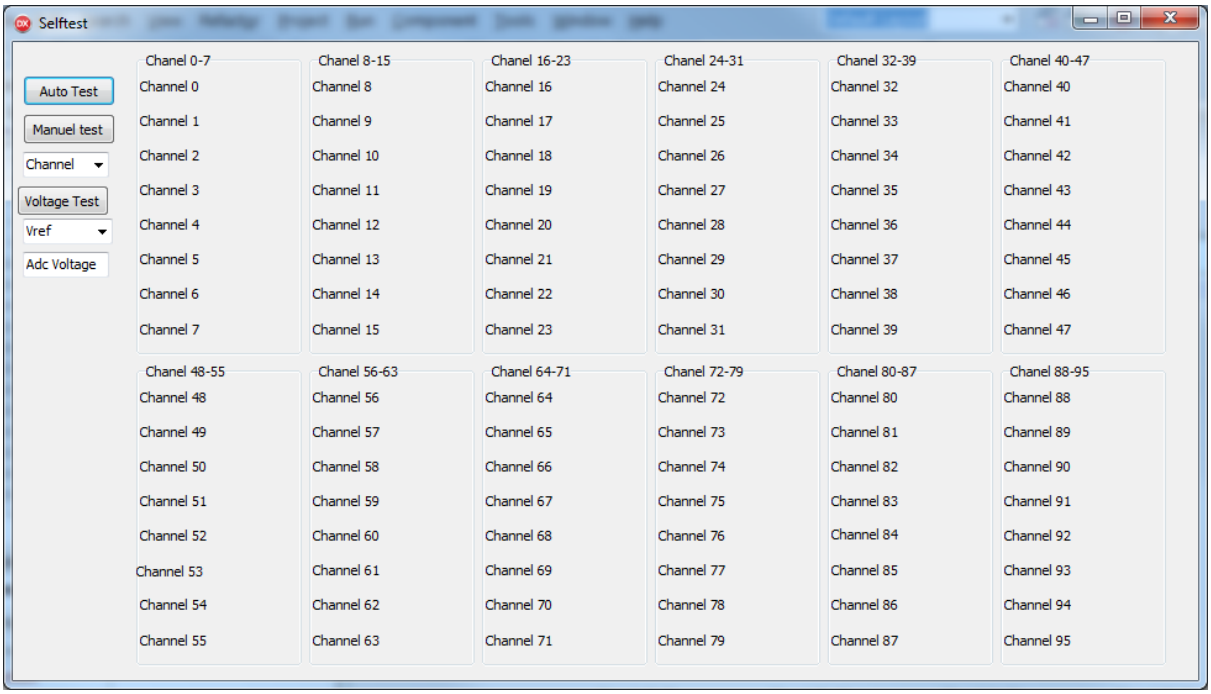


FIGURE 35 GUI SELF-TEST PROGRAM

The GUI of the self-test program shown in figure 35 is almost the same as the main GUI. The difference is that not the ADC module will be tested but the ADC tester itself. With the ADC on board of the tester the voltage reference can be checked.

The self-test includes the test of the 96 individual switches with the I^2c bus expander as well.

4.5.2. The Hardware

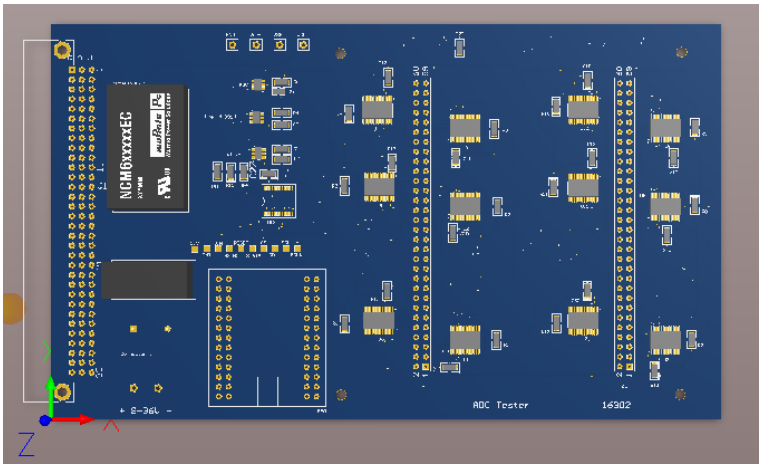


FIGURE 36 3D MODEL ALTIUM

In figure 36 a model of the final design of the hardware is shown. The PCB board design is in appendix 3

4.6. Realization phase

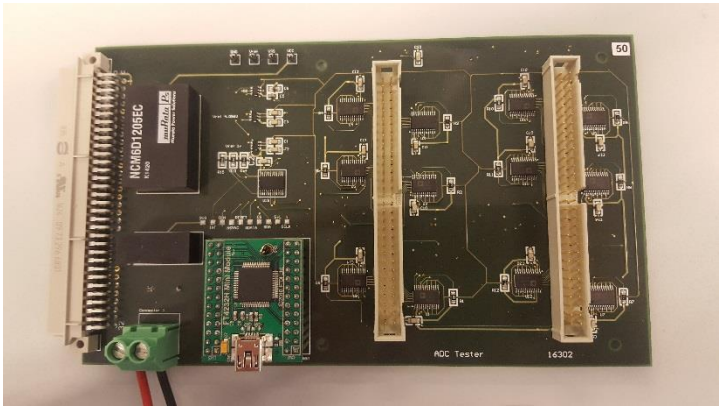


FIGURE 37 ADC TESTER

In figure 37 a picture of the finished ADC tester prototype is shown. To test the functions of the ADC tester, a test setup was build. A picture of the test setup is shown in Figure 38.

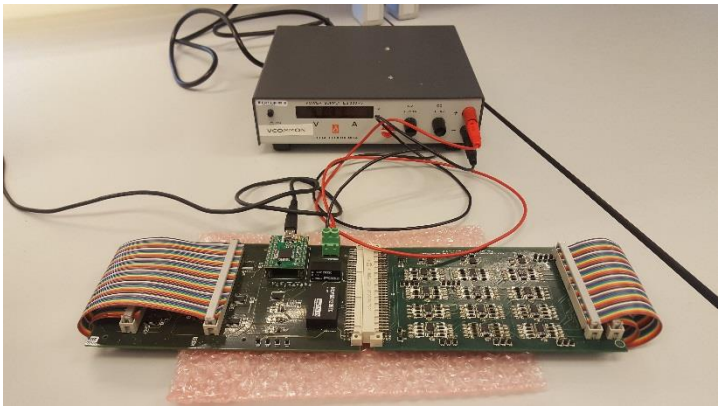


FIGURE 38 TEST SETUP

The test setup is connected with the computer via the USB connector. The written software is tested on the test setup.

5. Discussion

In the discussion all the questions are answered. The main questions was: Which hardware and software is needed to test the ADC module of the EVA setup without the help of the EVA setup? To be able to answer the main question first the sub questions needed to be answered. The sub questions are divided in the phases of the SDM method. The first phase of the SDM method is the problem confrontation phase. The questions in this phase are stated beneath including their answer.

- What is an ADC?
  - An ADC is an Analog to Digital converter.
- Which ADC is on the EVA plugin board?
  - The ADC on the ADC board is the AD7927. The AD7927 is a 12-bit, high speed, low power, 8-channel, successive approximation ADC.
- What is Fan out?
  - Fan out is defined as the maximum number of inputs (load) that can be connected to the output of a gate without degrading the normal operation.
- What is bit banging?
  - Bit banging is a technique for serial communications using software instead of dedicated hardware.

After the sub questions from the problem confrontation phase were answered, the problem analyst phase was conducted. The sub questions of the problem analyst phase are stated beneath including their answer.

- How does the ADC on the Eva board work?
  - The ADC on the EVA board is a successive approximation ADC.
- On what voltage does the ADC operate?
  - The ADC operates on 3,3 Volt and 5 Volt.
- What needs to be measured?
  - The outcome of the ADC to see if it is still working correctly.

After the problem analyst phase the functional design phase / definition phase was conducted. The sub questions of this phase are stated beneath including their answer.

- How to control the ADC
  - The ADC can be controlled via I^2C.
- How to control the ADC tester?
  - The ADC tester can be controlled with use of the software written during this thesis project.
- What signals will the tester use?
  - The tester is using I^2C.

After the design phase / definition phase was executed, the physical phase was conducted. The sub questions of this phase are stated beneath including their answer.

Physical design phase

- How to connect the ADC module to the ADC tester?
  - Via Flat cables.

After answering all the sub questions the main question can be answered. By executing the different phases of the SDM method the hardware and software was realized to test the ADC module without the help of the EVA test setup.

6. Conclusion

The main question of the thesis project was: With what hardware and software can the ADC module of the EVA setup be tested without the Eva setup? The research which was conducted resulted in a standalone ADC tester for the ADC module of the EVA test setup.

During the design of the ADC tester the research was split into two parts, namely the hardware and software.

The hardware part is designed with the following requirements

- Standalone (without the EVA set up)
- Self-test function
- Needs test pads to measure signals
- Connect to USB
- Connect to ADC module
- Wrong power connection protection a diode in the + of the connection

The design of the hardware is based on these requirements.

For the self-test function seven I^2C bus expanders are placed. One of the I^2C bus expanders is used for controlling the components on the ADC tester. The other six are used as inputs to test the analog switches. For every signal that is used is on the ADC tester a test pad is placed so it can be easily measured with an oscilloscope. For connection the ADC to USB an FT4232 mini module is chosen because NXP had experience with this module and it can control an I^2C bus. To be able to test all the channels individually on the ADC module 96 analog switches were chosen. The ADG 1414 houses eight analog switches so 12 of them are placed. The ADC tester uses two different voltage references: a 1,025 Volt and 4,096 Volt reference to be able to test the ADC module on two different levels. The voltage references can be switched on and off via the software. The 4.096 voltage reference is also used for the self-test function of the ADC tester. If there is 4,096 Volt on the input of the I^2C expander it will register as a '1' and so the analog switches can be tested. To make sure the voltage references are working correctly an ADC is placed on the ADC tester so the voltages can be checked.

It was a challenge to take all these requirements into account. The different components needed to create the ADC tester had to be researched and chosen based on the specifications of the datasheet. Some of the components were chosen because of NXP was already using them and had a positive experience with them.

The software is designed with the following requirements:

- GUI needs to be easy in use
- The ADC tester needs to be able to test the 96 channels individually.
- Manual Testing
- Automatic testing

The software is designed with in mind that someone with limited knowledge can still operate the ADC tester. The software created is therefore made to be logical and functional. There is no further knowledge needed to use the software. By making the GUI simple in by only having 4 main functions. So the GUI is easy in use.

To conclude the phases, the main question needs answering: With what hardware and software can the ADC module of the EVA setup be tested without the Eva setup? All the steps taken in het report lead to the answer: With the designed hardware and software the ADC module of the EVA setup can be tested. The chosen hardware is specifically chosen to create an ADC tester. All the components are matched so they can work together.

7. Recommendations

The hardware and software of the ADC tester function both however there is always room for improvement.

During testing a couple of design mistakes where discovered. The addresses of the IO expanders were not in order. The next step would be a redesign so the IO expander addresses will be in order.

To reduce the costs, the DC-DC converter can be changed to a smaller one. The current DC-DC converter is chosen for max load on the switches. However, the switches will never be at full load.

The power supply connector is placed backwards in the PCB design. On the design the text + 9-36V – is visible but in the prototype the connector is covering this text see figure 39. My recommendation is to flip the connector in the design so it can be placed without covering the text.

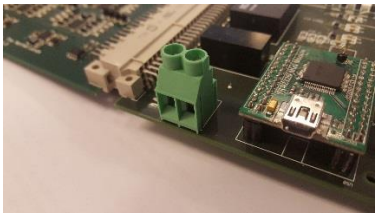


FIGURE 39 POWER SUPPLY CONNECTOR

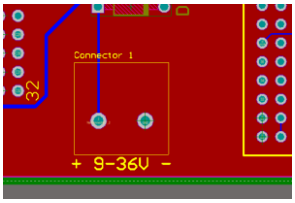


FIGURE 40 PCB POWER SUPPLY CONNECTOR

Sometimes the ADC board is not working, in that case a reset is needed by disconnecting the board from the tester. The problems with the ADC board are already know by NXP. To understand this problem further research is needed on the ADC module.

Works Cited

A Dictionary of Computing. (2017, 1 4). Retrieved from encyclopedia.com: http://www.encyclopedia.com/computing/dictionaries-thesauruses-pictures-and-press-releases/nassi-shneiderman-chart

ADC. (2016, 11 03). Retrieved from Farnell: http://www.farnell.com/datasheets/1789968.pdf?_ga=1.68410770.2015399304.1472808944

Bit banging. (2016, 11 8). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Bit_banging

Bredius, M. (2011). *DFT Design Specification TJA1022 Doc Rev 0.1 draft — 23 Feb 2011 internal document.*

DAC. (2016, 11 03). Retrieved from Farnell: http://www.farnell.com/datasheets/1487349.pdf?_ga=1.173808928.2015399304.1472808944

difference-between-fan-in-and-fan-out-in-digital-electronics. (2016, 11 8). Retrieved from verticalhorizons: http://verticalhorizons.in/difference-between-fan-in-and-fan-out-in-digital-electronics/

digital-analog-conversion. (2016, 11 03). Retrieved from allaboutcircuits: http://www.allaboutcircuits.com/textbook/digital/chpt-13/digital-analog-conversion/

electronics-tutorials. (2017, 1 5). *electronics-tutorials.* Retrieved from sequential: http://www.electronics-tutorials.ws/sequential/seq_5.html

i2c-tutorial. (2016, 11 02). Retrieved from www.robot-electronics.co.u: http://www.robot-electronics.co.uk/i2c-tutorial

Intro to LIN. (2016, 10 18). Retrieved from Newark: http://www.newark.com/pdfs/techarticles/introToLIN.pdf

Malik, B. (2016, 11 2). *analog-to-digital-ADC-converter*. Retrieved from microcontrollerslab: http://microcontrollerslab.com/analog-to-digital-ADC-converter-working/

NXP. (2016, 10 31). Retrieved from NXP: http://www.nxp.com/wcm_documents/about/pdfs/factsheet.pdf

robots.ox.ac.uk. (2016, 12 15). Retrieved from background_lectures: http://www.robots.ox.ac.uk/~gari/teaching/b18/background_lectures/1P2-Op-Amp-Circuits-L3-Notes-Collins.pdf

TJA1022 Datasheet. (2016, 10 17). Retrieved from NXP: http://www.nxp.com/products/interface-and-connectivity/wired-connectivity/can-lin-flexray-transceivers/lin-transceivers/dual-lin-2.2a-sae-j2602-transceiver:TJA1022

8. Appendices

Appendix 1 ADC Tester Schematic

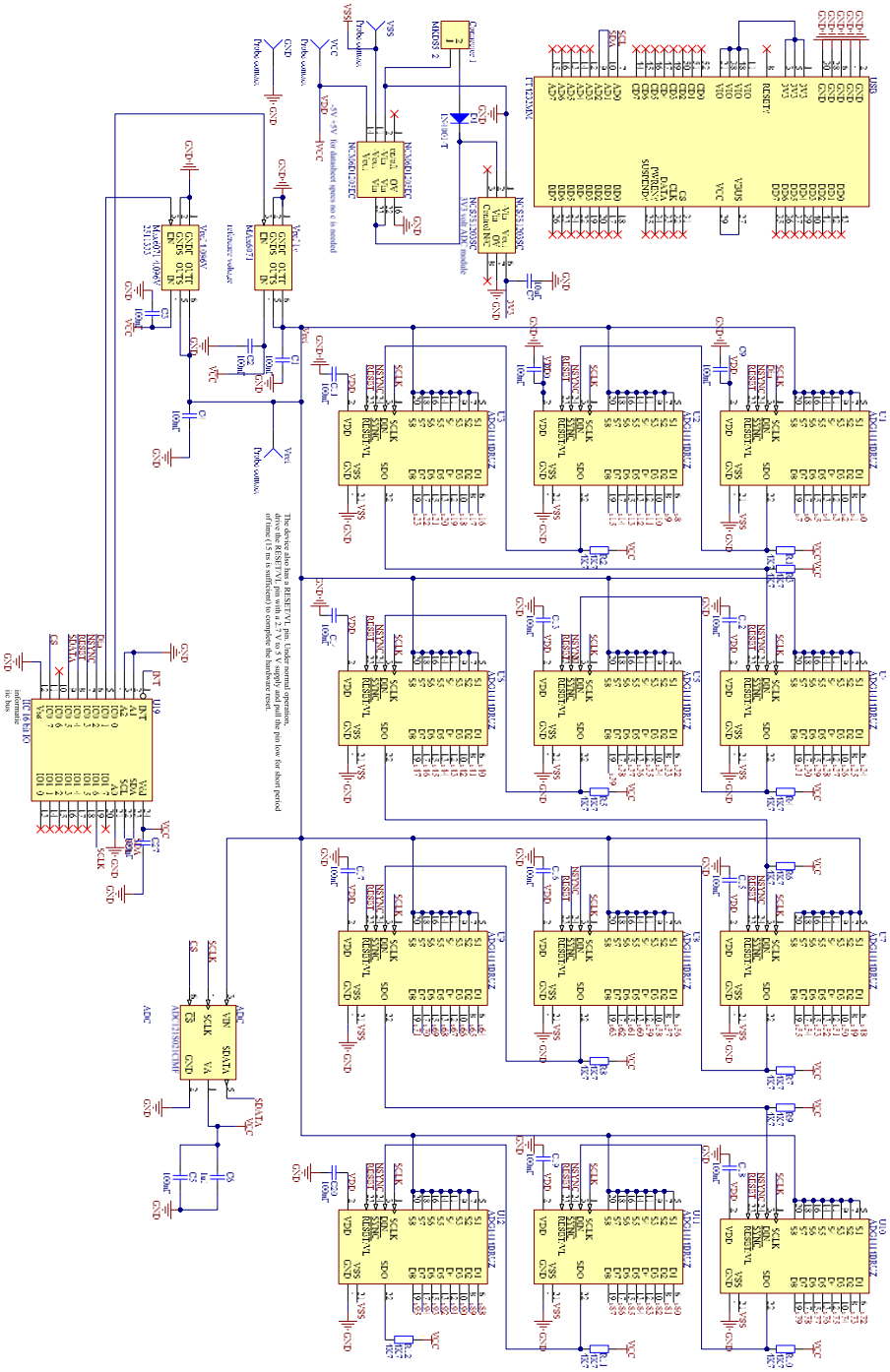


FIGURE 41 ADC SCHEMATIC

Appendix 2 ADC Tester Schematic 2

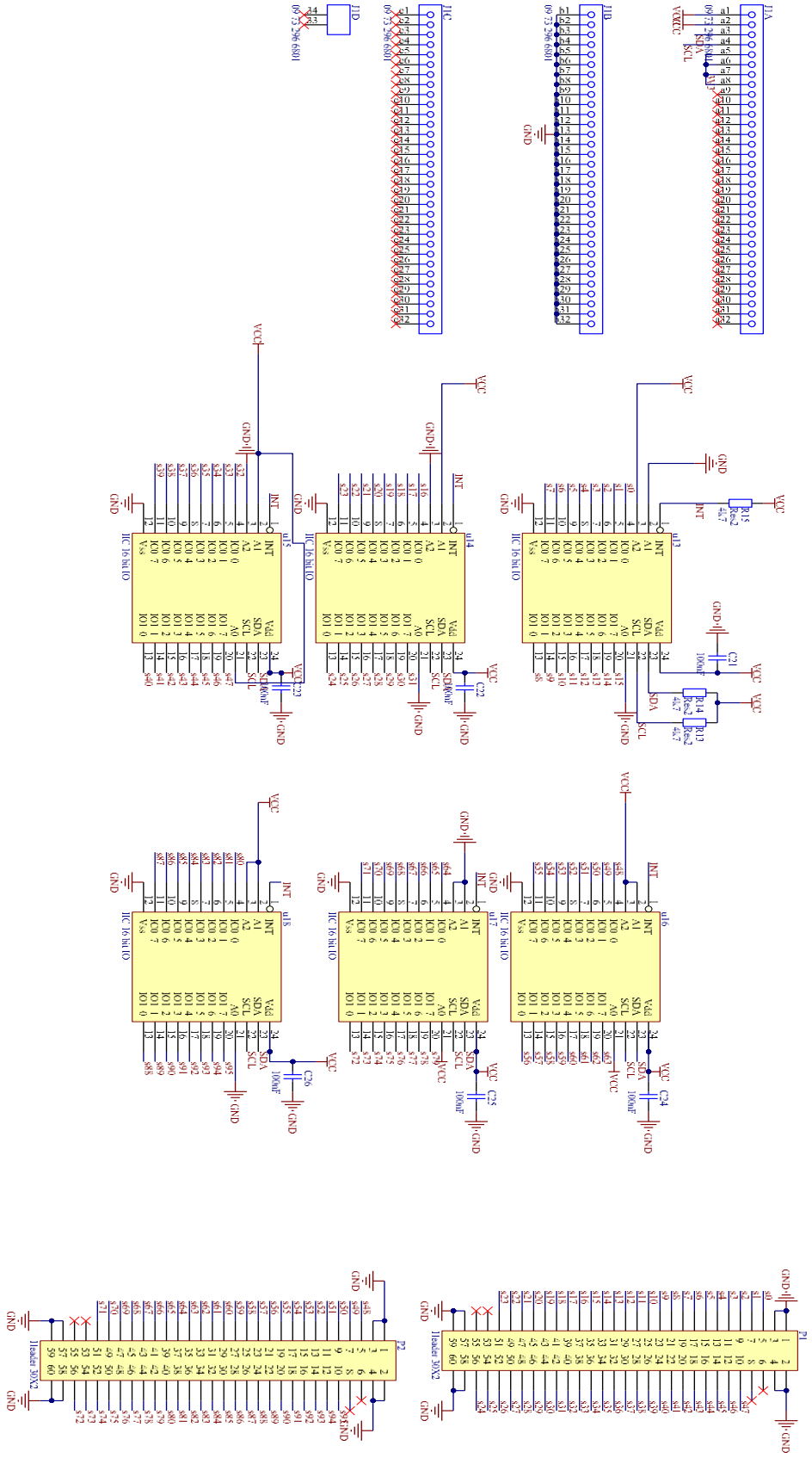


FIGURE 42 ADC SCHEMATIC 2

Appendix 3 ADC PCB

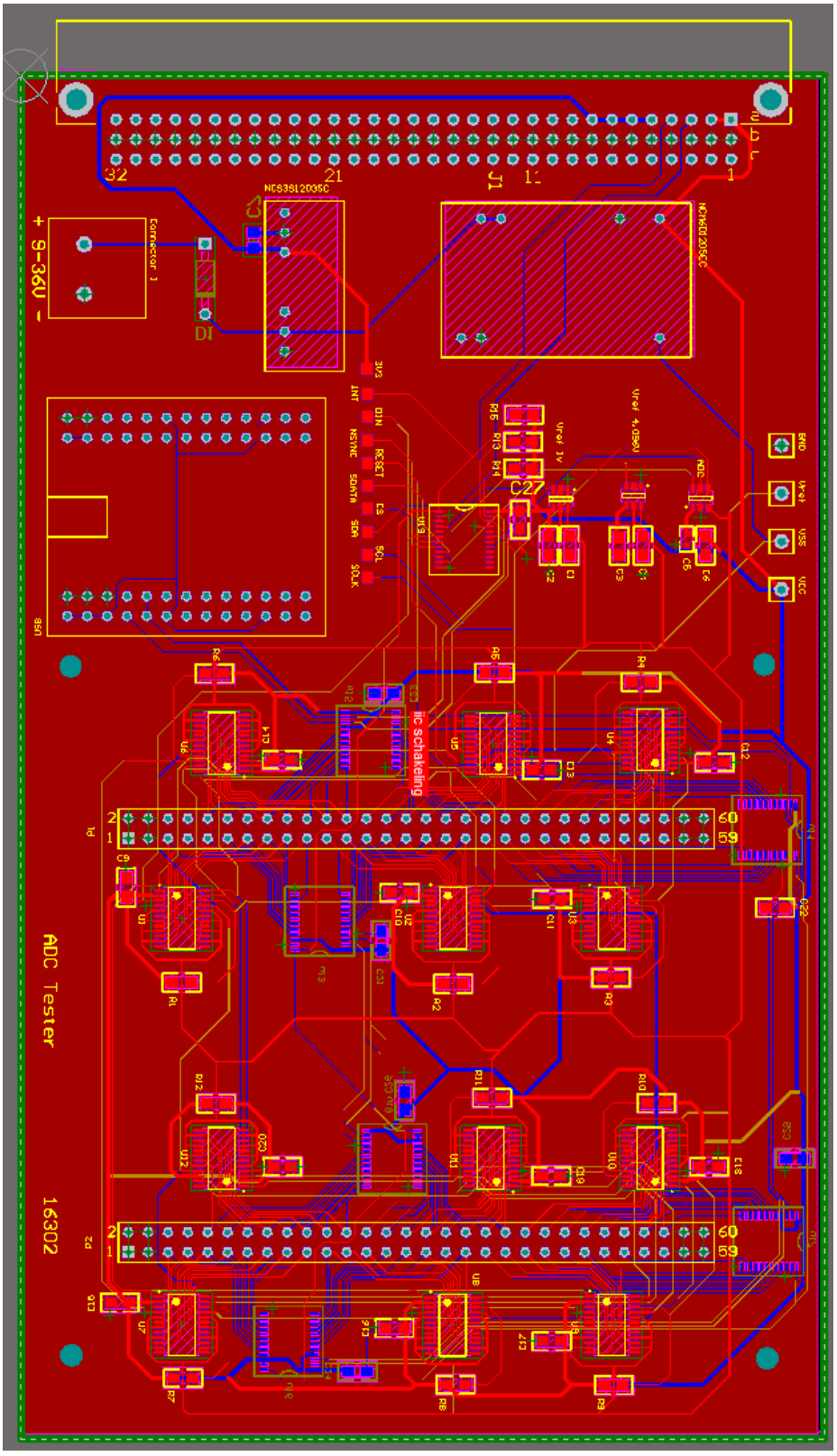


FIGURE 43 ADC TESTER PCB

Appendix 4 Code of the program.

Procedure Initialization

With this code the whole ADC tester is Initialized

```
procedure init;
begin
  Set_Start;
Send_Byte_Then_Chk_ACK($40);          //1    set on output
Send_Byte_Then_Chk_ACK($06);           register
send_Byte_Then_Chk_ACK($00);           set the pin on off
send_Byte_Then_Chk_ACK($00);
  Set_Stop ;

    Set_Start;
Send_Byte_Then_Chk_ACK($40);          //1    IO expander 1
Send_Byte_Then_Chk_ACK($06);
send_Byte_Then_Chk_ACK($00);
send_Byte_Then_Chk_ACK($00);
  Set_Stop ;

  Set_Start;
Send_Byte_Then_Chk_ACK($42);          //2 IO expander 2
Send_Byte_Then_Chk_ACK($06);
send_Byte_Then_Chk_ACK($FF);           set pin on input
send_Byte_Then_Chk_ACK($FF);
  Set_Stop ;

  Set_Start;
Send_Byte_Then_Chk_ACK($44);           //3 IO expander 3
Send_Byte_Then_Chk_ACK($06);
send_Byte_Then_Chk_ACK($FF);
send_Byte_Then_Chk_ACK($FF);
  Set_Stop ;
  Set_Start;
Send_Byte_Then_Chk_ACK($46);              //4 IO expander 4
Send_Byte_Then_Chk_ACK($06);
send_Byte_Then_Chk_ACK($FF);
send_Byte_Then_Chk_ACK($FF);
  Set_Stop ;
  Set_Start;
Send_Byte_Then_Chk_ACK($48);               //5 IO expander 5
Send_Byte_Then_Chk_ACK($06);
send_Byte_Then_Chk_ACK($FF);
send_Byte_Then_Chk_ACK($FF);
  Set_Stop ;

  Set_Start;
Send_Byte_Then_Chk_ACK($4E);               //6 IO expander 6
Send_Byte_Then_Chk_ACK($06);
```

```
send_Byte_Then_Chk_ACK($FF);
send_Byte_Then_Chk_ACK($FF);
 Set_Stop ;

 Set_Start;
Send_Byte_Then_Chk_ACK($4C);                //7 IO expander 7
Send_Byte_Then_Chk_ACK($06);
send_Byte_Then_Chk_ACK($FF);
send_Byte_Then_Chk_ACK($FF);
 Set_Stop ;
end;
```

Procedure Voltage Reference

```
procedure TForm1.VrefChange(Sender: TObject);

begin
case vref.itemindex of       //       the index of the combo box
0:begin                          //    starting at the right value
 Set_Start;
Send_Byte_Then_Chk_ACK($40);
Send_Byte_Then_Chk_ACK($02);
send_Byte_Then_Chk_ACK($09); // set voltage 1,25 Volt
send_Byte_Then_Chk_ACK($00);
 Set_Stop ;
end;
1:begin
 Set_Start;
Send_Byte_Then_Chk_ACK($40);
Send_Byte_Then_Chk_ACK($02);
send_Byte_Then_Chk_ACK($0A); // set voltage 4 Volt
send_Byte_Then_Chk_ACK($00);
  Set_Stop ;
end;
2:begin
  Set_Start;
Send_Byte_Then_Chk_ACK($40);
Send_Byte_Then_Chk_ACK($02);
send_Byte_Then_Chk_ACK($00); // voltage ref off
send_Byte_Then_Chk_ACK($00);
  Set_Stop ;
end;
end;
end;
```

shiftregisterfill

```
procedure shiftregisterfill(chan:integer);
var
  I: Integer; with the variable chan the channel of the switch can be selected.
begin
   for I := 0 to 95 do
   begin
   if chan=i then
   begin
     din:=$4;
  end
   else
   begin
   din:=$0;
   end ;
   Set Stop ;

   end;
end;
```