Technical information

ISD

IPD7
Martijn de Groot
Edwin Kempen
Luuk Wolters

Content

1. Introduction

This document is all about the technical information there is in this product. Every sub part of the product is explained. At the end of the document is a system overview. This document also includes a test report. The system is fully tested.

2. System requirement and specifications

2.1Requirements

In this document is specified the requirements and specifications of our project. The first part exists out of requirements. These were specified in accordance with our client.

Functional requirements:

- Input is conform MIDI protocol.
- The signal is translated to Light and vibration.
- LEDs are used instead of normal bulbs.
- We need multiple LEDs for strong light output.
- We want to generate vibration with vibrating motors.
- The First prototype is based on a basic drum set up.
- From the DD-65 kit one is used

Not functional requirements:

- The client wants to use the prototype during individual lessons and group lessons.
- The controller needs to be as universal as possible.
- The LED module is based on the Yamaha DD-65. The module needs to have the same lay-out.

2.2 Specifications

The specifications consist of two parts. The First part are the written specs, the second part is a schematic overview view.

Our system is build of the following parts:

- LED module + controller

  The LED module converts sound to light.

  The controller is used for reading the MIDI signal, storing it and to translate it into usable output. This output is used by the LED module and used by the vibration module.

- Vibrato module

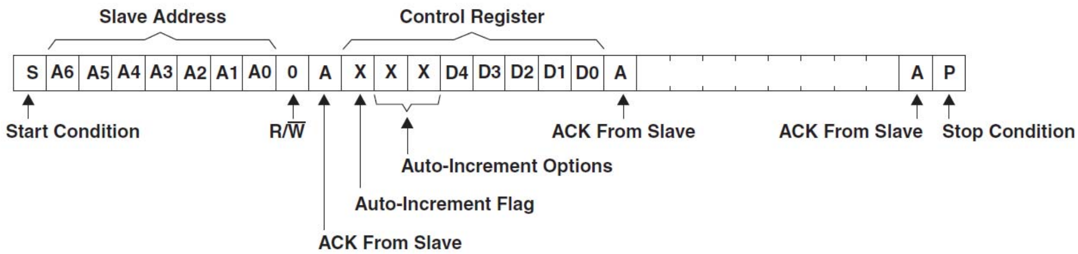
  The vibrato module converts sound to feeling, with use of vibrations.

Input

- Signal according to General MIDI protocol
- Bytes that are used:
  - Channel
  - Note
  - Velocity
- Generated by a drum computer, in test setup by dd-65.
- Every drum kit could be used.
- If the learn button is pushed, ISD will learn which code belong to which pad.

Treatment

- The treatment of the signal is done with an ATmega128 microcontroller.
- MIDI signal is stored in the memory of the microcontroller.
- The signal is stored just long enough to work with it.
- The signal is split into the three bytes that are needed.
- Microcontroller output to the LED driver is in accordance with the I2C-protocol.
- I2C command exist of :



- LED driver output is in PWM configuration.

- LED driver drives the power drivers with the PWM-signals.
- The PWM-signal has a value from 0 – 100% and with steps of 10%.
- Power driver sends the amplified PWM signal to
- Power driver drives the LED module en vibrato module.
- One signal is connected to multiple power driver inputs. This to get the required power.
- Working voltage for the microcontroller and LED driver: 5V dc. Power driver does not require power supply.

Output

- LED module
  - 6 different color LEDs.
  - Distance between LED and outside has to be 8 cm.
  - There are 9 pads that need to be lid.
  - Double color for hi-hat pad.
  - The different colored LEDs have different working Voltages.

- Vibrato module
  - Needs to be carried around the abdominal.
  - The vibrators are placed on different places for different drums.
  - The vibrating motors are in different sizes.
  - The vibrating belt could be disconnected when not used.

Power supply:

- Power supply is bought in.
- Different output Voltages: 12V and 5V.
- Peek power of 120W.
- Peek current of 10A.

3 I2C LED Driver TLC59116

The microcontroller output goes through an I2C bus. This bus is connected to the TLC59116. This driver converts the code to PWM signals. This driver could deliver 16 independent PWM signals.

The TLC59116 has some control registers that control the PWM signals. The control registers are controlled by software code. They are generated by the microcontroller.

It is possible to use multiple LED drivers in the same I2C bus. There for you need to give the device an address. In the setting we use the address is all put to a logic one. This means that de address lines are connected to the vcc with a pull up resistor.

To prevent that driver is over powered; the output is set to a maximal current. This current is set to 5,6mA. For limiting the current there is a value for Rext needed. The following formula will take care of this: $Iout = \frac{1,25}{Rext} \times 15 => Rext = 3k3$

With this setting the maximal current will be 5,6mA.

4 LED current source

The I2C LED driver couldn't supply the power to drive the LEDs. That's why we need the have some kind of amplifier. This is why we design a voltage controlled current source. The drive signal is a PWM signal that is converted to a current setting.

As you could see in the schematic it's a simple circuit. The circuit is made around two pnp transistors. This need to be pnp transistors because de driver only works in sink mode. The bc557 is a small signal transistor. This transistor drives the power transistor. The power transistor is BD140 and is heavy enough for our purpose.

Between the base of the bc557 and the +12V is a zener diode. Together with resistor connected to the emitter of the BD140, it will generate the current settings. In this setting the circuit wouldn't deliver more than 600mA. This is a safe value for the 3W power LEDs that are used.

The circuit could have been simplified. It is possible to use only a resistor to get the current settings. But if this is used the resistor needed is a 7W resistor. This is very big and not as accurate like our circuit. In our circuit the resistor is just 1W.

At the base of the bc557 is also a resistor. This is for transistor settings. The maximal current that will flow to the tlc59116 I2C LED driver is 1mA. The TLC59116 is limited at this value.

5 The MIDI to Serial converter.

This circuit is placed between the input of the MIDI signal and the serial uart input of the ATMega128. The function of this circuit is to convert a MIDI signal to a signal on TTL level. The MIDI signal is a differential signal. This means it has no reference to ground, but a reference to each other. A microcontroller cannot handle such signal, which is why this circuit is between the MIDI input and the microcontroller.

The serial output is build with a pull-up resistor to VCC. This means the line is hold to VCC for as long no data is transmitted. When MIDI data comes into the circuit, the serial output varies between GND and VCC. For the microcontroller, VCC level stands for a logic “0”, while GND level stands for a logic “1”

Another function of this circuit is to create galvanic isolation between the MIDI device and the Interactive Silent Drums. This means that the Interactive Silent Drums cannot damage the connected MIDI
device.

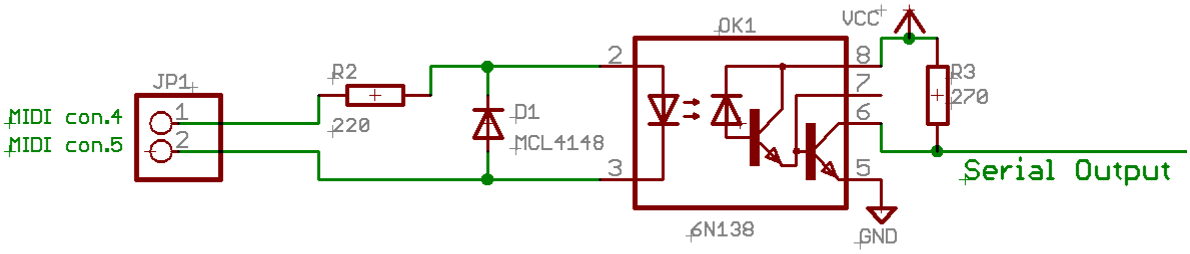


Figure 1 Schematic of the MIDI to serial converter used in the Interactive Silent Drums

6 Vibration belt

The drummer can feel vibrations by using micro drives. This are miniature vibrating motors that are placed in a belt. The Interactive Silent Drums uses five different micro drives, of the company Precision Micro drives. The micro drives get their power out of the current driver circuit as earlier discussed in this document.
To get the correct current RX is calculated depending on the micro drive, see figure 2. The values of the other parts in the circuit are not depending on the used micro drive.

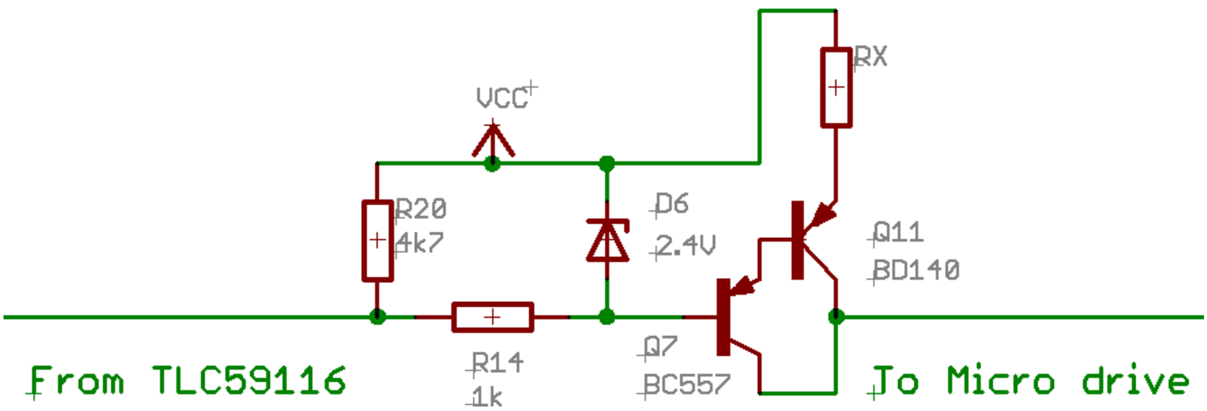


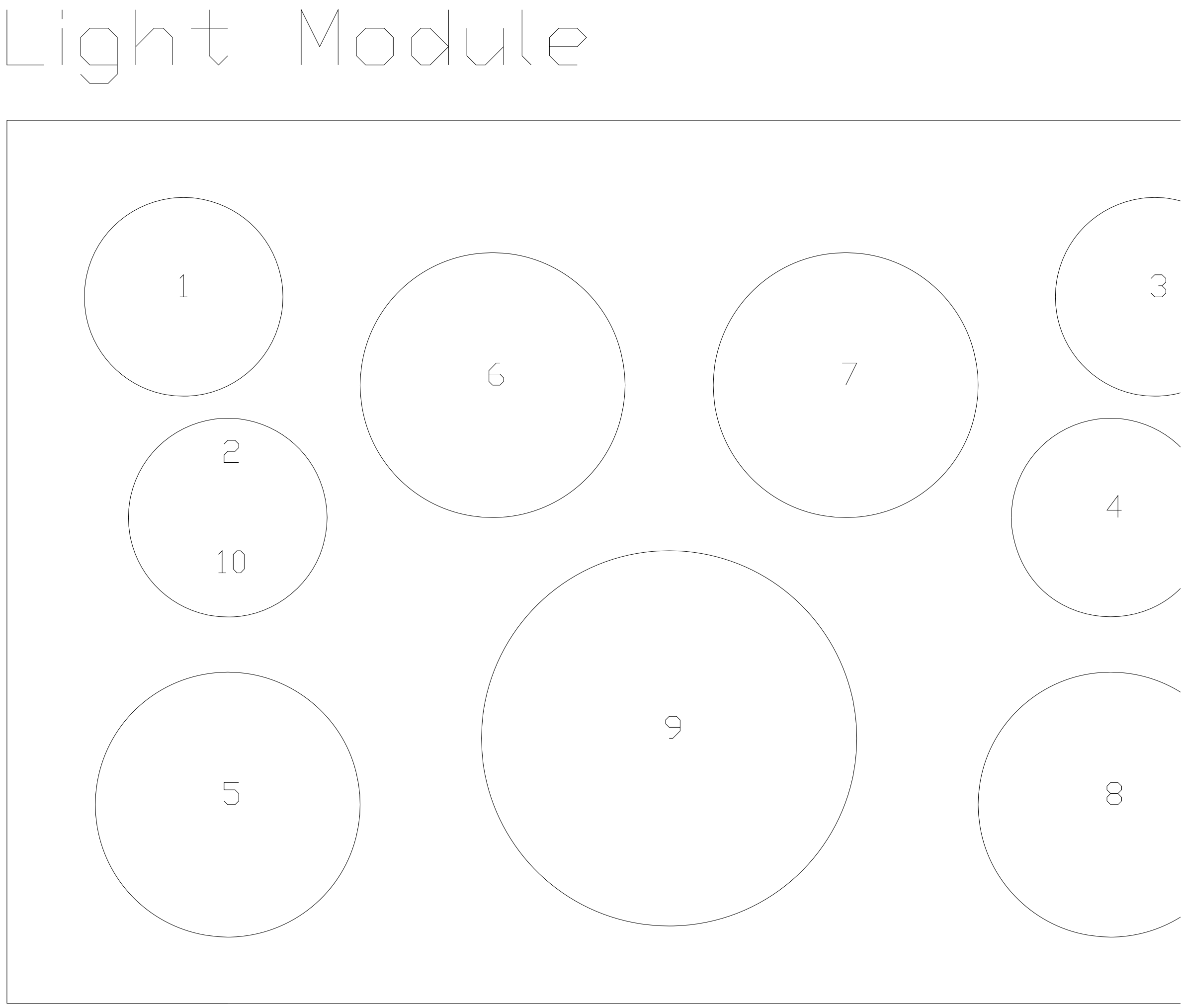
Figure 2 Current driver to power the micro drives

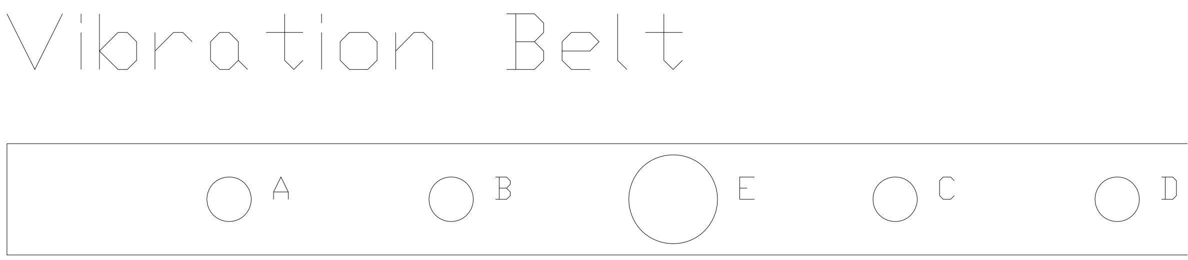
Vibration belt		Technical Parameters			Cable colour		Connector pin	
Drive Name	Description	Micro drive	Current [mA]	Resistor	Supply	GND	Supply	GND
A	Pad 1	308-100	100	8R2	Red	Black	1	9
B	Pad 2	310-103	110	5R6	Green	White	2	10
C	Pad 3	310-101	85	10R	Yellow	Blue	3	11
D	Pad 4	312-101	120	10R	Purple	Brown	4	12
E	Kick	318-001	260	2R7	Orange	“Shield”	5	13

Table 1 Overview of the details of the micro drives used in the Interactive Silent Drums.

7 Lay out

Figure 5 shows the layout of the Interactive Silent Drums, which is based on Yamaha DD65 electrical drum kit. Table 5 shows detailed information about the lay out. The table shows which light and vibration module corresponds with a MIDI note number. The MIDI channel is 9.



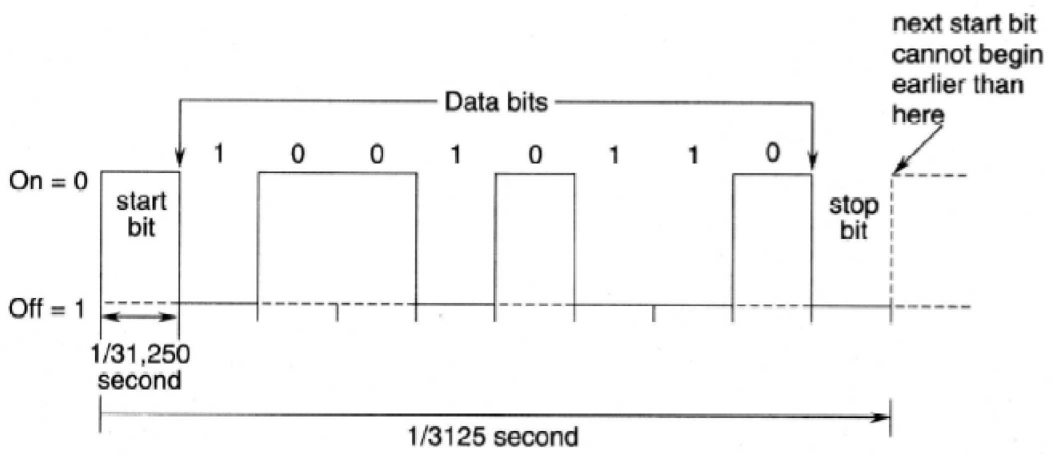


			MIDI	
Light Module	Vibration belt	Instrument	Note#	Note
1		Crash cymbal1	49	C# 2
2		Hi-Hat Closed Power	42	F# 1
3		Ride Cymbal 2	59	B 2
4		Splash Cymbal	55	G 2
5	A	Snare M	38	D 1
6	B	High Tom	50	D2
7	C	Mid Tom L	47	B 1
8	D	Floor Tom H	43	G 1
9	E	Bass Drum	36	C 1
10		Hi-Hat Pedal	44	G# 1

8 Input interface

The communication we use for the Interactive Silent Drums is MIDI. We have chosen this option for 2 reasons. The first reason is that you can use midi with any drum set (even analogue ones) and the second reason is that it`s a very easy and understandable bus protocol.

The main idea behind midi is that every digital instrument or analogue one with triggers can send the music data to a computer. The music data consists of 3 bytes. The first byte represents the channel and the action that has to be made for example when our drum set is sending music the first byte is 0x99 (channel 9 with a note on event). The second byte represents the note number. With this note number you know which pad has been hit. The third byte represents the velocity this means how hard you hit the pad. With this information we know how bright the corresponding led has to light up. The byte information is send with a start and stop bit. You can use the normal uart interface on a micro controller to capture the midi bytes. The baud rate of this signal is 31250 Baud (this means



31250 bits per second). Between every byte there is a pause bit.

9 Microcontroller

We used a micro controller to capture the midi signals. We calculated the value for the baud rate register UBRR= ((16 MHz/ (16*31250 Baud))-1= 31(hexadecimal). After you set the frame the frame consists out of 1 start bit 8 data bits and 1 stop bit with no flow control and parity check.

The bytes that midi is generating are stored in the UDR register. The only thing we do is check the udr register and check if there is a note on command and after this which pad (note number) is hit and how hard they hit it (the velocity). With this information we can generate a specific pwm signal. The only problem now is that we have to generate 15 PWM signals. When the normal controller has to generate 15 pwm signals in software, because hardware pwm is not possible for such an amount of pwm signals, he misses a lot of midi bytes and this is something you do not want. We decided for this reason that we are going to use an extra IC that provides 16 independent pwm signals with an i2c protocol.

The micro controller we used was an atmega 128 cause this a micro controller for a reasonable price and with a lot of options (round 8 euro`s). A big advantage for the project group was that the knowledge about this microcontroller was already there. The code for this micro controller was due to this reason already finished in about 2 months.

The only the micro controller does is:

1 set the i2c chip

2 set the uart

3 wait on the midi signals

4 when you know which pad has been hit set the correct register for the pwm chip

5 set the velocity*2 in this register

6 send it over i2c and execute it in the chip

7 led will light up and when you hit the low tones the belt is going to vibrate

The micro controller itself needs to have a 16 MHz oscillator and 100nF capacitors to remove distortion from the power line. For the coupling between midi and the uart interface for the micro controller we are using an optocoupler. When something happens we have a galvanic separation between the signals and off course a beautiful ttl signal to the microcontroller.

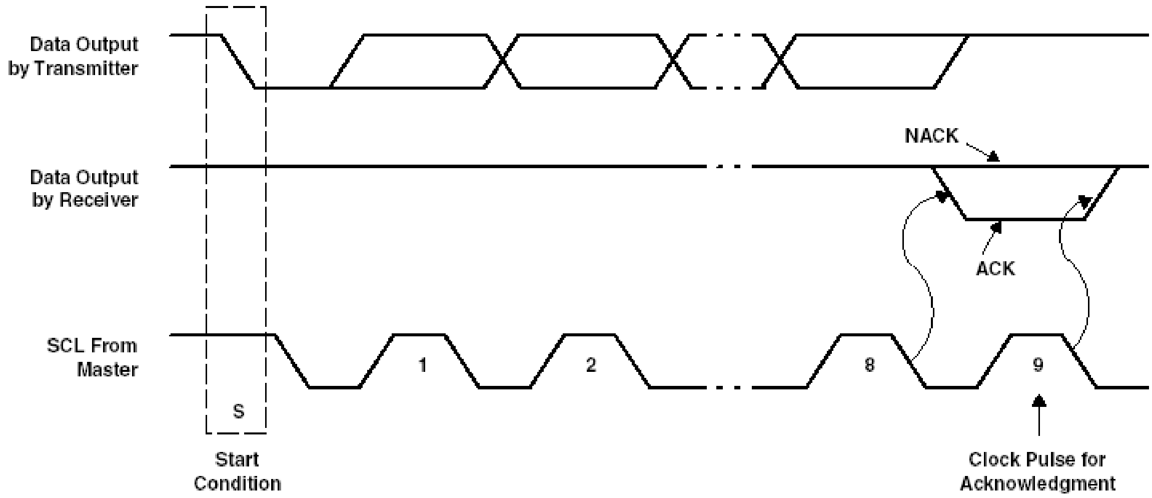


Figure 14. Acknowledge/Not Acknowledge on I^2C Bus

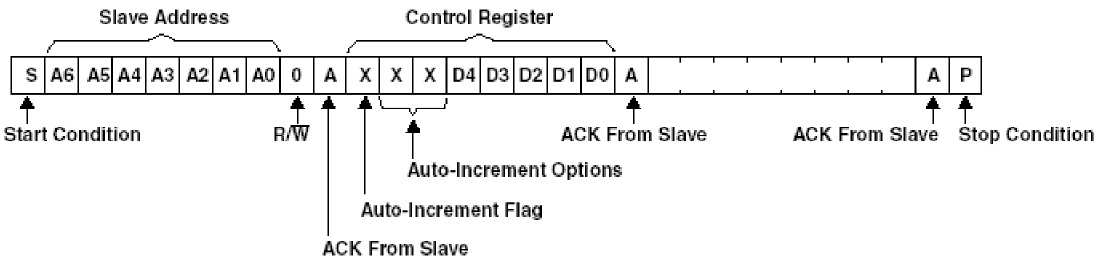


Figure 15. Write to a Specific Register

10. Introduction of test report

This test report gives an impression of the tests that are carried out, to make sure that the Interactive Silent Drums work according to the specifications. The report also contains images that are shot from the oscilloscope. This is to prove the correct measurements on the device.

10.1. Functional testing of the ISD

10.1.1. Tools

The following tools where used during the test of the ISD.

- Oscilloscope (Fluke 99B)
- Multi meter

10.1.2. Test 1: Power Supply

This is the first test that has to pass, before continuing the further tests. This is because a correct functionality of the ISD is not guaranteed when this test fails.
Test form 1 shows the measure points on the controller PCB with their specified voltages and their measured voltages. Test form 2 shows the measure points on the LED driver PCB.
The voltages are measured with an oscilloscope.

10.1.3. Test 2: MIDI to Serial converter

This second test is to make sure that IC1 (ATMega128) on the controller PCB, receives the data as is specified. First a MIDI signal is measured on pin 2 and 3 of the OK1 (6N138 optocoupler). This is a differential voltage. Figure 2.1 show the measured signal. This shows a signal with an amplitude of 2,5V. This means the MIDI signal reaches the input of OK1.

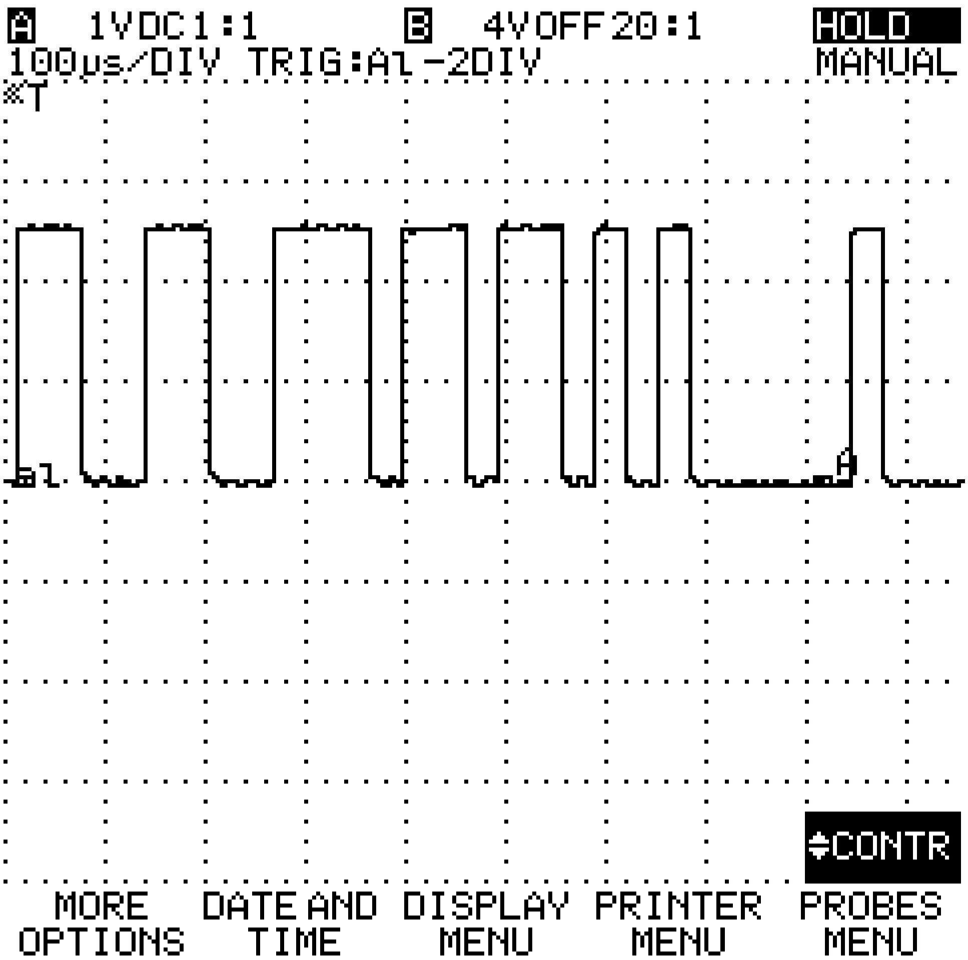


Figure 2.3.1The MIDI input of the 6N138 optocoupler.

The output of OK1 shows a serial bit pattern as specified. The signal is now +5V or 0V to ground. Figure 2.2 shows a bit pattern measured on the RXD input of the ATMega128. .

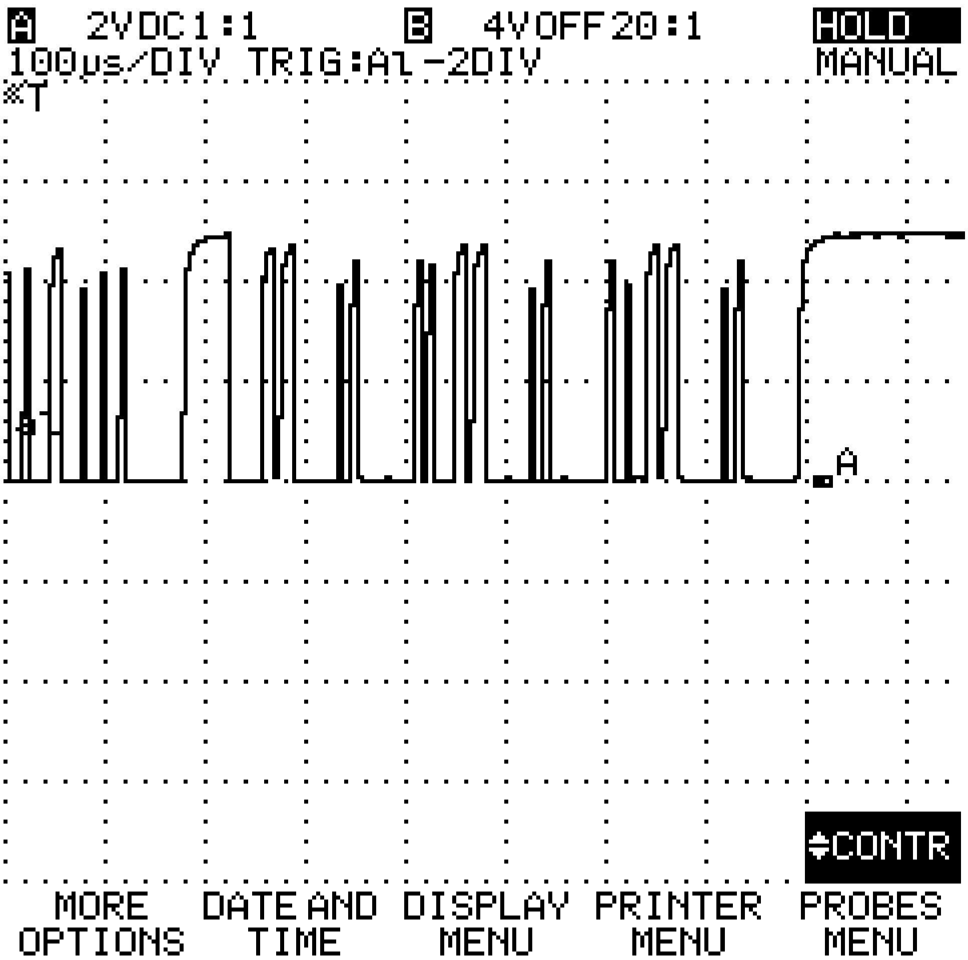


Figure 2.3.1 The RXD input of the ATMega128.

Test 2: Passed.

10.1.4. Test 3: The clock frequency

To test the correct functionality of the ATMega128, test 1 and 2 had to been passed. Another important input of the ATMega128 is the clock frequency. The clock frequency of the crystal is tested with an oscilloscope. Figure 2.4.1 shows the measured signal. According to the specifications, the signal is a sine wave with a frequency of 16 MHz.

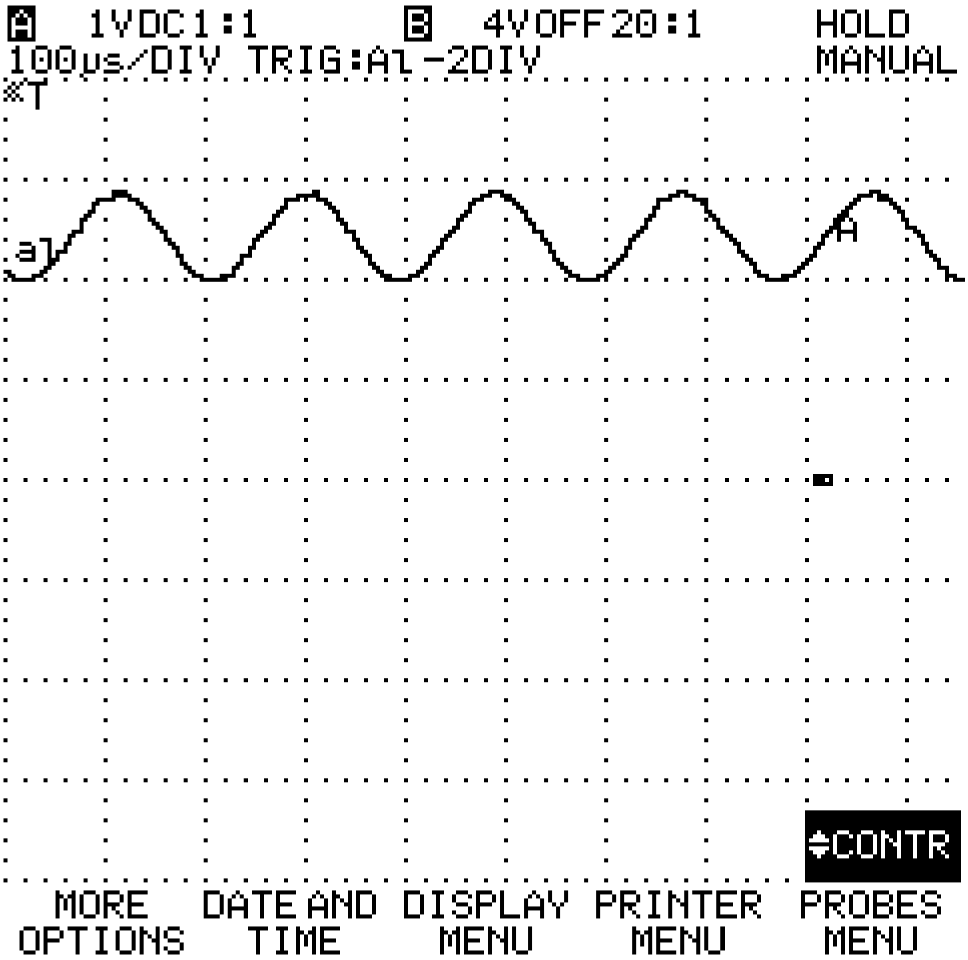


Figure 2.4.1. The output of the crystal.

<u>Test 3: Passed</u>

10.1.5. Test 4: I2C Clock signal

The I2C clock signal is detectable when data is transmitted over the I2C bus. The signal is measured with an oscilloscope. Figure 2.5.1 shows the clock signal. The clock has a frequency of 100 kHz.

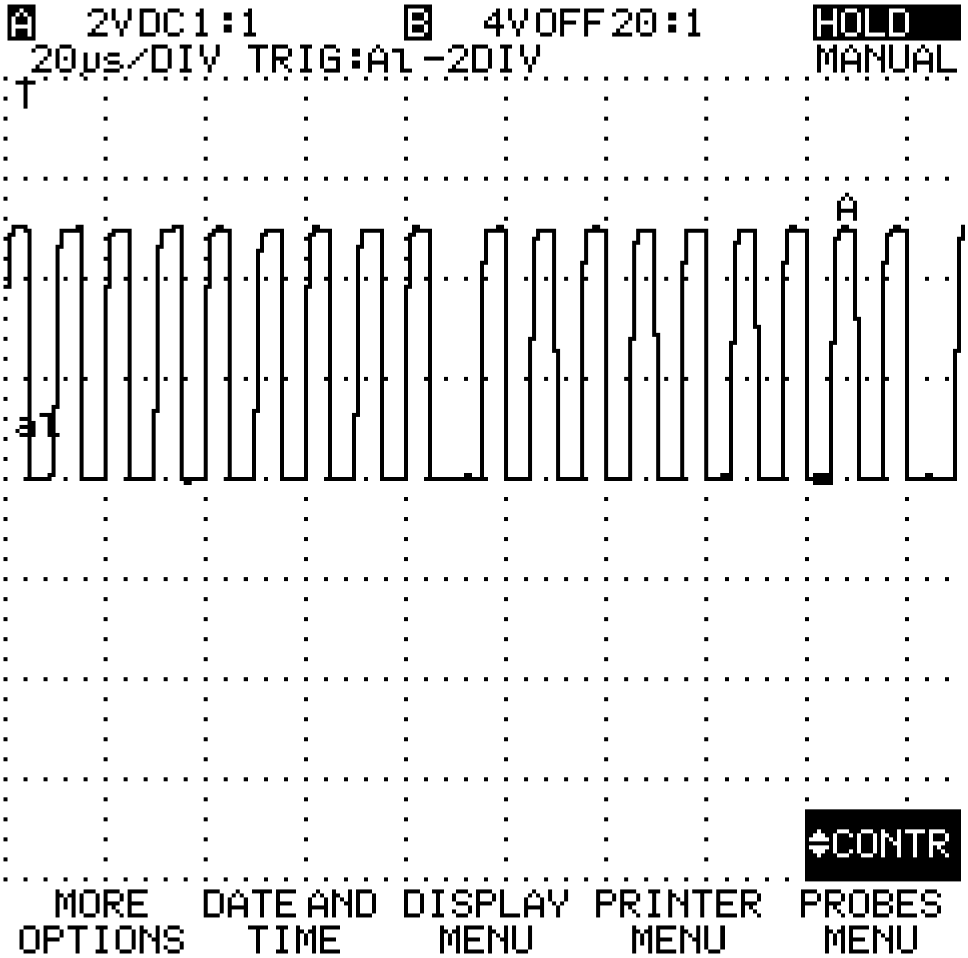


Figure 2.5.1. The I2C clock signal measured on the SCL output of the ATMega128

Test 4: Passed.

10.1.6. Test 5: I2C Data signal

The I2C data signal is detectable when data is transmitted over the I2C bus. The signal is measured with an oscilloscope. Figure 2.6.1 shows the measured signal.

Important: If the bus has a constant value of 5V, the bus is not working correctly.

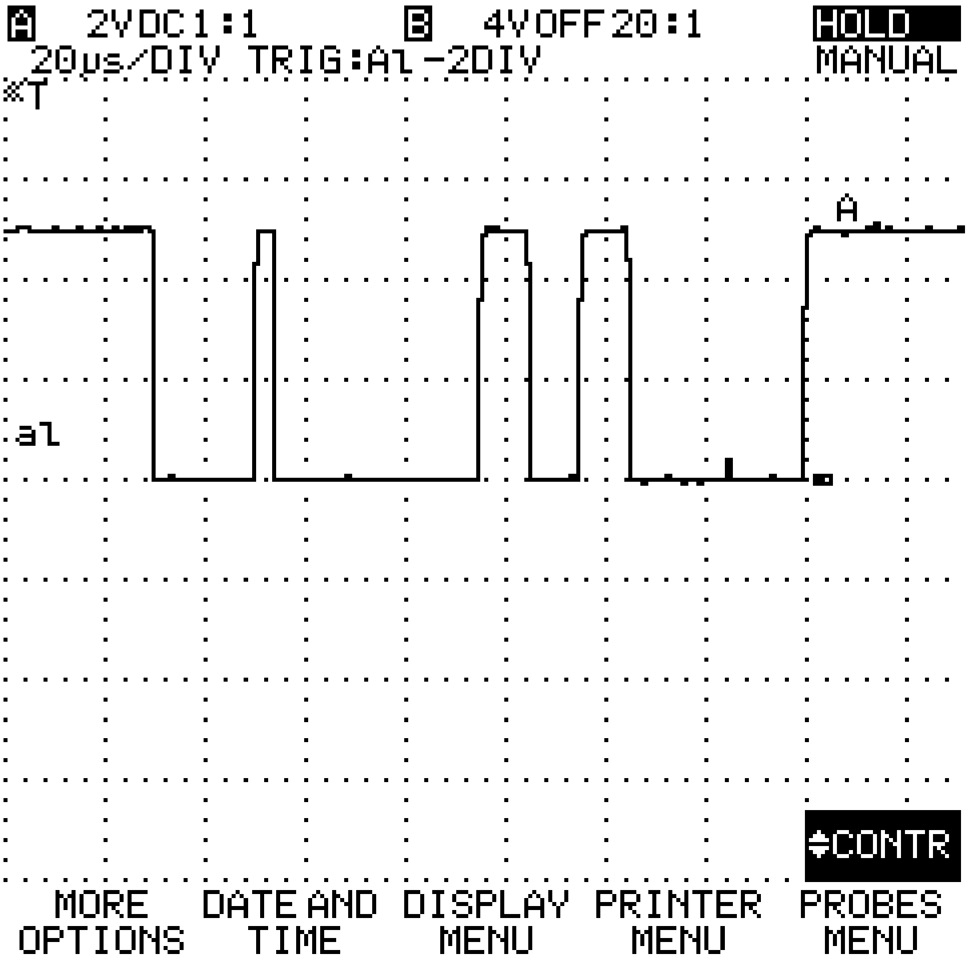


Figure 2.6.1 The I2C data signal measured on the SDA output of the ATMega128.

Test 5: Passed.

10.1.7. Test 6: PWM Signals out of the TLC59116

For this test a MIDI signal is transmitted to the ISD with a specified velocity. To guaranty a correct working of the TLC59116, four different velocities are tested. First we start with a velocity of 100%. Figure 2.7.1- figure 2.7.4 shows the signals measured on the output of the TLC59116. Notice that the voltages are inverted. This is because the TLC59116 switches to ground.

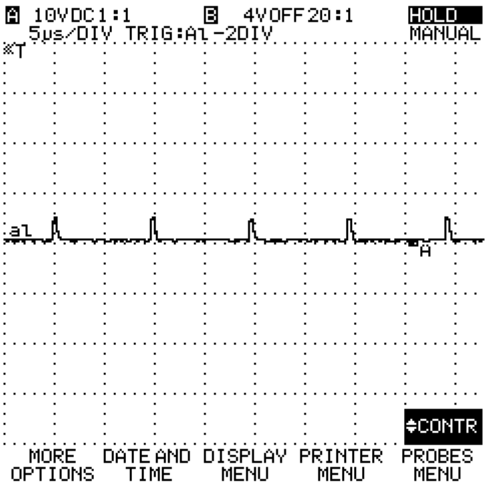


Figure 2.7.1 PWM when velocity is 100%

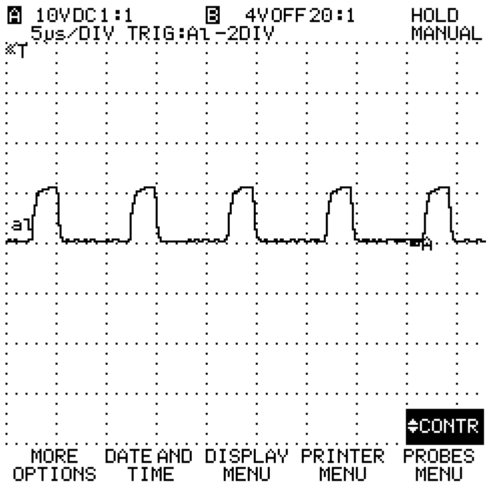


Figure 2.7.2 PWM when velocity is 75%

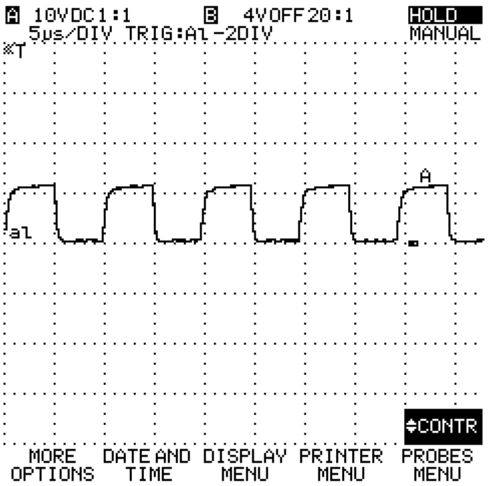


Figure 2.7.3 PWM when velocity is 50%

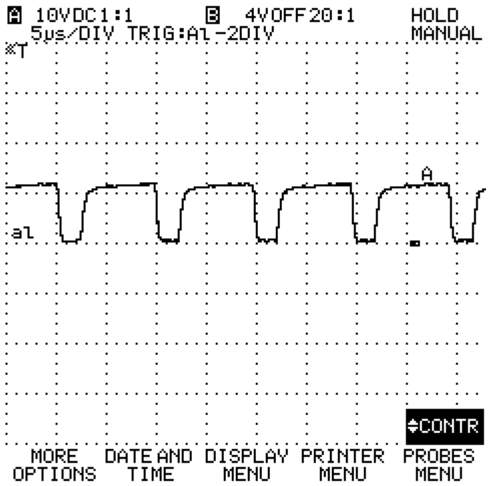


Figure 2.7.4 PWM when velocity is 25%

Test 6: Passed

10.2. Test Conclusion

The tests are carefully executed with the highest concentration. This is all being done so that the functionality of the product is guaranteed, according to the specifications. All the tests came out with a good result according to the specifications.

Test form 1: Power Supply Controller PCB

					Voltage	
PCB	Device	Description	Pin	Description	Specified	Measured
Controller PCB	OK1	Opt coupler	8		5	5
	IC1	ATMega128	1	PEN	5	5
			21	VCC	5	5
			52	VCC	5	5
	Resistor		R1	VCC	5	5
	IC2	TLC59116	28	VCC	5	5
	Resistor		R21	VCC	5	5
	Resistor		R7	VCC	5	5
	Resistor		R22	VCC	5	5
	Resistor		R10	VCC	5	5
	Resistor		R12	VCC	5	5
	Resistor	Motor Driver	R16	VCC	5	5
	Resistor		D2	VCC	5	5
	Resistor		R15	VCC	5	5
	Resistor		R17	VCC	5	5
	Resistor		D3	VCC	5	5
	Resistor		R25	VCC	5	5
	Resistor		R18	VCC	5	5
	Resistor		D4	VCC	5	5
	Resistor		R26	VCC	5	5
	Resistor		R19	VCC	5	5
	Resistor		D5	VCC	5	5
	Resistor		R27	VCC	5	5
	Resistor		R20	VCC	5	5
	Resistor		D6	VCC	5	5
	Resistor		R28	VCC	5	5

Name test engineer: Edwin Kempen

Date: 02-02-2009

Test form 2: Power Supply Led Driver PCB

					Voltage	
PCB	Device	Description	Pin	Description	Specified	Measured
LED power PCB	Resistor	LED Driver	R1	+12V	12	12
	Resistor		D1	+12V	12	12
	Resistor		R2	+12V	12	12
	Resistor		R4	+12V	12	12
	Resistor		D2	+12V	12	12
	Resistor		R6	+12V	12	12
	Resistor		R7	+12V	12	12
	Resistor		D3	+12V	12	12
	Resistor		R9	+12V	12	12
	Resistor		R10	+12V	12	12
	Resistor		D4	+12V	12	12
	Resistor		R12	+12V	12	12
	Resistor		R25	+12V	12	12
	Resistor		D9	+12V	12	12
	Resistor		R27	+12V	12	12
	Resistor		R13	+12V	12	12
	Resistor		D5	+12V	12	12
	Resistor		R15	+12V	12	12
	Resistor		R16	+12V	12	12
	Resistor		D6	+12V	12	12
	Resistor		R17	+12V	12	12
	Resistor		R20	+12V	12	12
	Resistor		D7	+12V	12	12
	Resistor		R19	+12V	12	12
	Resistor		R22	+12V	12	12
	Resistor		D8	+12V	12	12
	Resistor		R23	+12V	12	12
	Resistor		R28	+12V	12	12
	Resistor		D10	+12V	12	12
	Resistor		R30	+12V	12	12

Name test engineer: Edwin Kempen

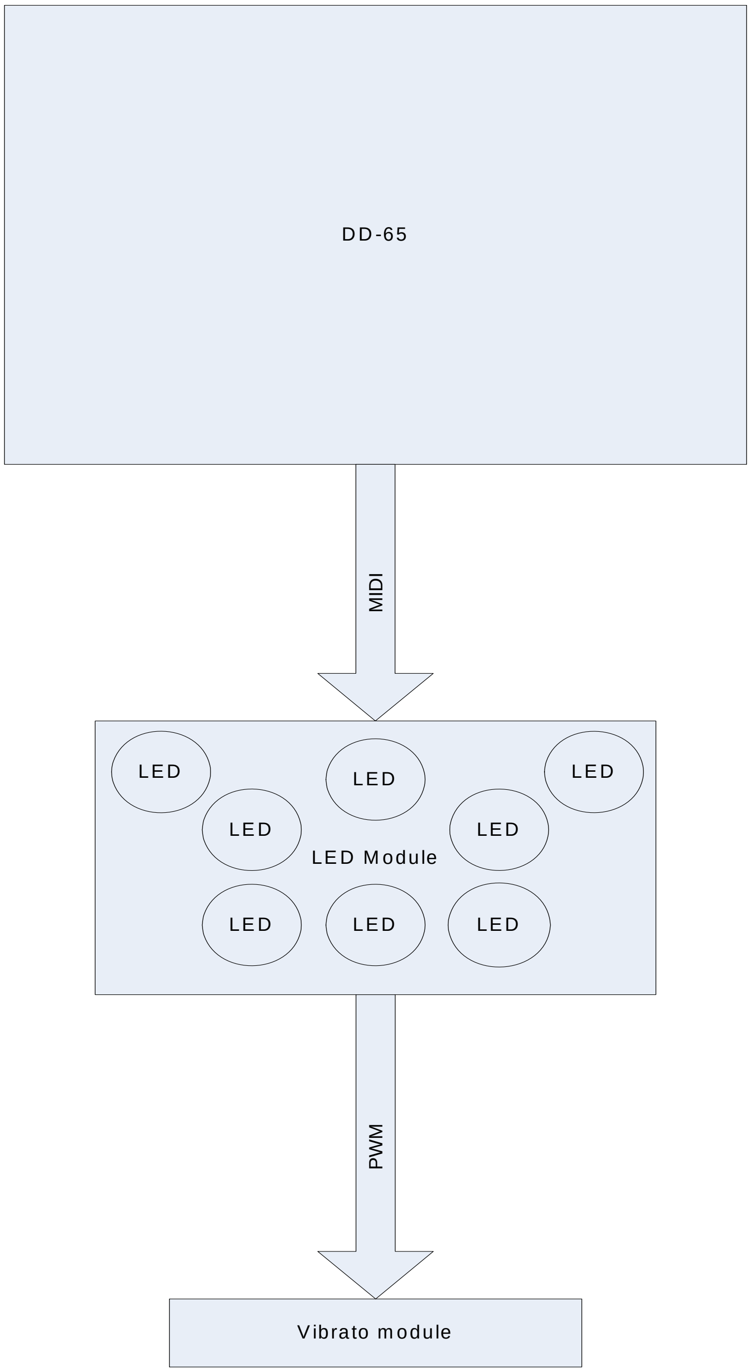
Date: 02-02-2009

11. System overview

Over the next three pages is a system overview.

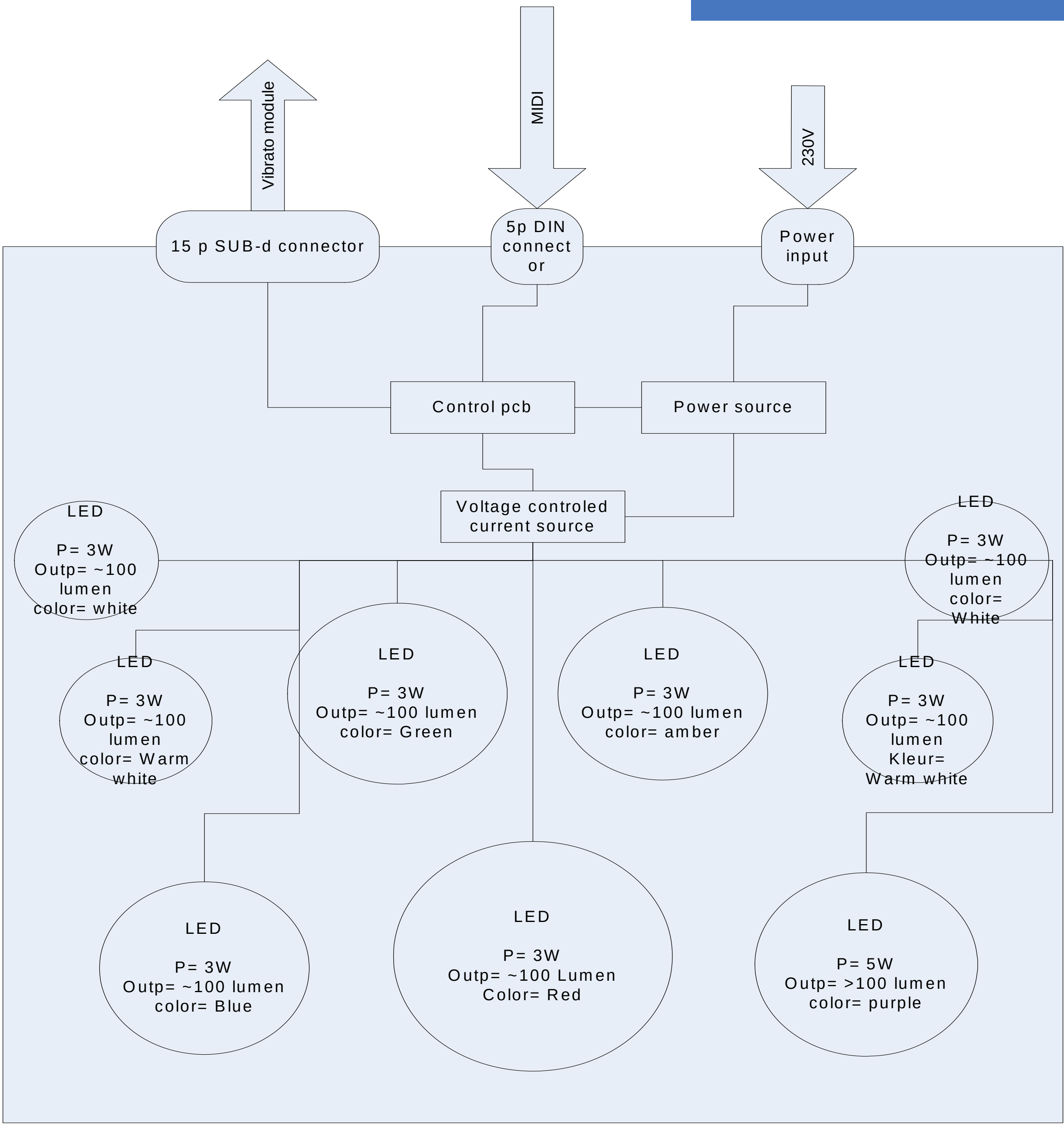
Totaal schema vaste module

donderdag 12 februari 2009



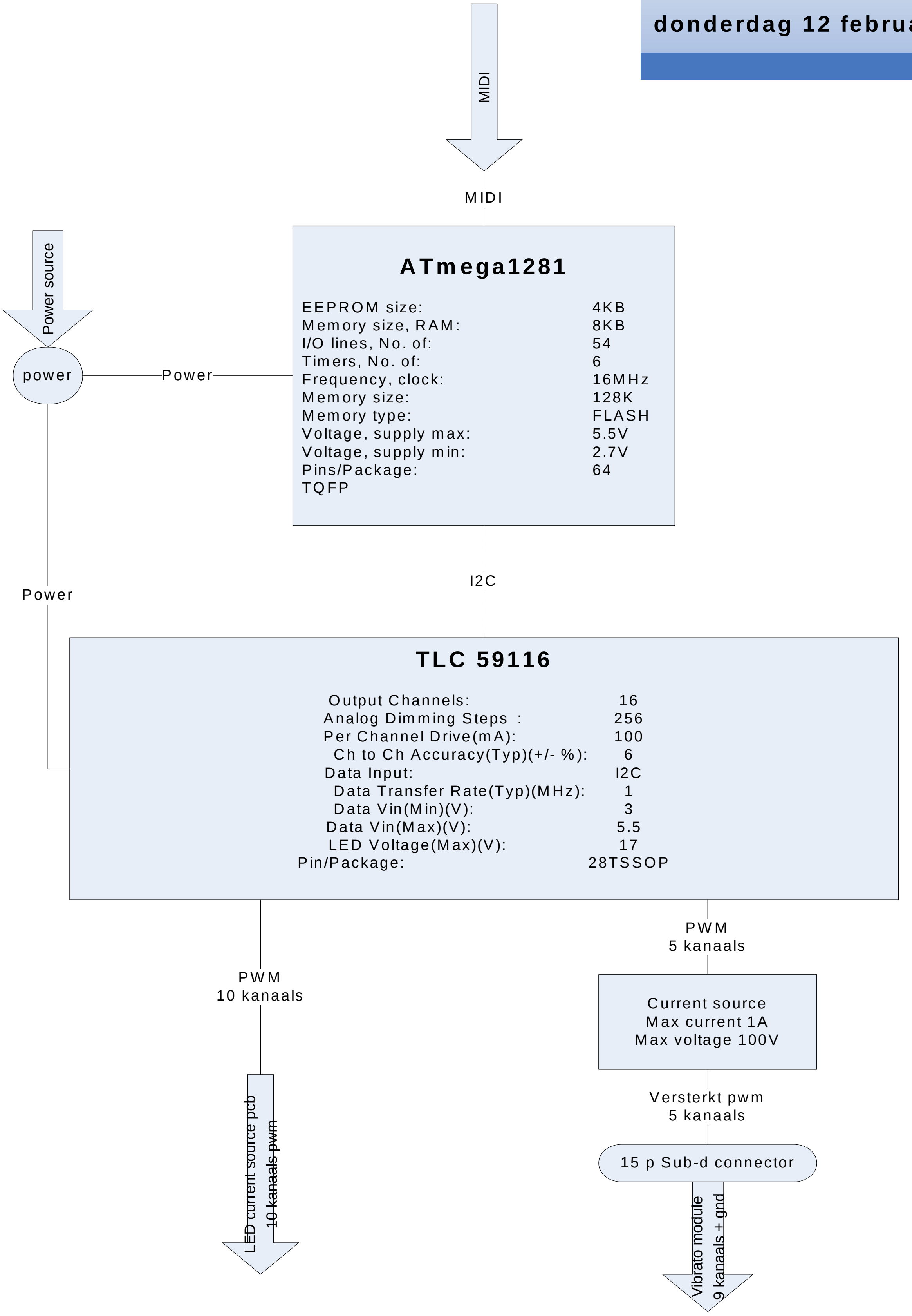
LED module

donderdag 12 februari 2009



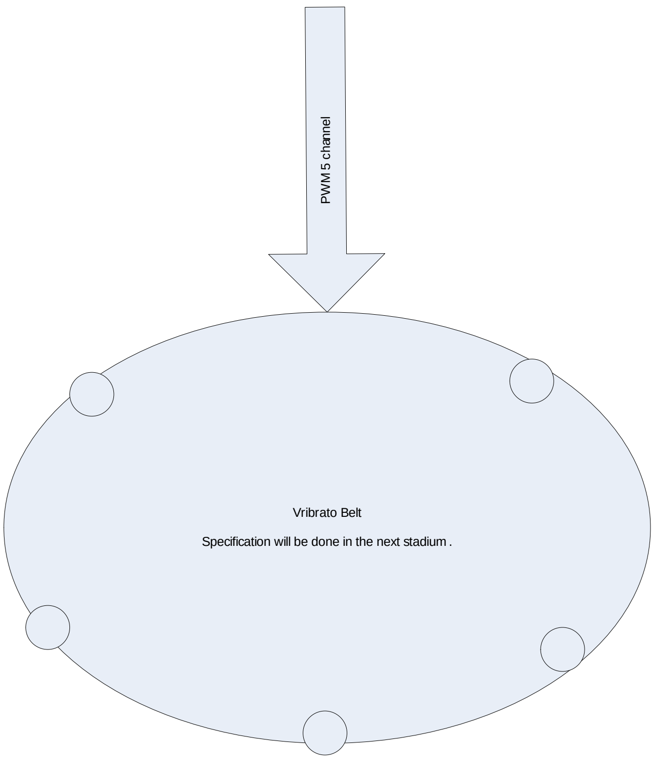
Control PCB

donderdag 12 februari 2009



Totaal schema vaste module

Thursday, February 12, 2009



Attechments