# Microcontroller programmeren

Inhoud

[Microcontroller programmeren 1](#_Toc192588074)

[1 Microcontrollers en programmeren 2](#_Toc192588075)

[1.1 voorbereiding 2](#_Toc192588076)

[1.2 practicum 5](#_Toc192588077)

[2 GPIO 6](#_Toc192588078)

[2.1 voorbereiding 6](#_Toc192588079)

[2.2 Practicum 6](#_Toc192588080)

[3 Interrupts 9](#_Toc192588081)

[3.1 Voorbereiding 9](#_Toc192588082)

[3.2 Practicum 10](#_Toc192588083)

[4 Clocks, Timers & PWM 13](#_Toc192588084)

[4.1 Voorbereiding 13](#_Toc192588085)

[4.2 Practicum 14](#_Toc192588086)

[5 ADC 15](#_Toc192588087)

[5.1 Voorbereiding 15](#_Toc192588088)

[5.2 practicum 15](#_Toc192588089)

## 1 Microcontrollers en programmeren

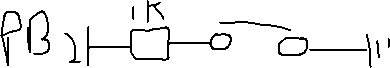
### voorbereiding

4. i/o staat voor input/output. Ermee word bedoet pins waar je data mee kan ontvangen of sturen. Peripherals zijn externe hardware waar data naar gestuurd kan worden of van kan worden ontvangen.
5. 1. ADCn: een chip die analoge naar digitale signalen omzet
   2. DACn: een chip die digitale naar analoge signalen omzet
   3. RTC: een klok
   4. TCAn: timer
6. 42
7. A screenshot of a computer program

   Description automatically generated
8. Input
9. PB3 / 7
10. De led heeft een 1k voorschakelweerstand en zit verbonden aan pb3. Als je de led naar ground verbind gaat hij aan



1. 0 want dan zit de led verbonden aan vcc
2. Pb2
3. Als je m indrukt gaat de i/o lijn naar gnd
4. Een restistor om te voorkomen dat de waarde gaat zweven



1. 0 want de i/o pin word verbonden aan gnd
2. Dat als je een knop indrukt dat hij eerst een aantal keer heel snel heen en weer gaat tussen aan en uit omdat het contact “stuiterd”
3. Met een condensator
4. A diagram of a machine

   Description automatically generated
   1. 7 segmenten display
   2. Oled display aansluiting
   3. Motor driver
   4. Knopjes
   5. Potmeter
   6. Sd kaart reader
   7. I2c connectie
   8. P1 connectie

|  |  |
| --- | --- |
| P1: 1 | Pb2 |
| 2 | Pf4 |
| 3 | Pf5 |
| 4 | Pa2 |
| 5 | Pa3 |
| 6 | Pa4 |
| 7 | Pa5 |
| 8 | Pa6 |
| 9 | Pa7 |
| 10 | Gnd |
| 11 | Pb0 |
| 12 | Pb1 |
| 13 | Pf2 |
| 14 | Pf3 |
| P2: 1 | vcc |
| 2 | Pd6 |
| 3 | Pd3 |
| 4 | Pd2 |
| 5 | Pd1 |
| 6 | Pd0 |
| 7 | Pd7 |
| 8 | Pd5 |
| 9 | Pd4 |
| 10 | Gnd |
| 11 | Pe3 |
| 12 | Pe2 |
| 13 | Pe1 |
| 14 | Pe0 |

|  |  |  |  |
| --- | --- | --- | --- |
| Component | Functie | Pin smu | Pin curiosity |
| Led1 | Indicatie led sd reader | P2.8 | Pd5 |
| Led2 | Indicatie led | P2.14 | Pe0 |
| Sw1 | Software defineerbare switch | P2.11 | Pe3 |
| Sw2 | Software defineerbare switch | P2.12 | Pe2 |
| Sw3 | Software defineerbare switch | P2.13 | Pe1 |
| pot | Potmeter te lezen met software | P2.9 | Pd4 |

1. Een korte pauze
2. \_delay\_ms(tijd in ms);

### practicum

2. Er moet “while”staan
3. Met dir an je aangeven of een pin input of output is  
   met out kan je een pin op hoog of laag zetten
4. Dir: initialisatie  
   out: loop
5. Dirset en outset
6. PORTE.DIR |= PIN0\_bm;  
   PORTE.OUT |= PIN0\_bm;
7. Outclr
8. Met een delay

## GPIO

### voorbereiding

1. 1. = maakt de variabele een bepaalde waarde
   2. ~ NOT operator. Draait een bit / bitreeks om.
   3. | OR operator. Als van 2 bitreeksen 1 van de 2 bits 1 is word het resultaat 1
   4. & AND operator. Alleen als van beide bitreeksen de bit 1 is word het resultaat 1
   5. ^ XOR operator. Alleen als 1 van de 2 bits uit een bitreeks 1 is word het resultaat 1
2. Dat zijn conditionele operatoren
3. 1111 1101
4. 1
5. 1100 1111
6. 1100 1101
7. 0000 0000
8. 1100 1101
9. 1
10. Or operator 1010 1010 | 0100 0000 🡪 1110 1010
11. XOR operator 1010 1010 ^ 1000 0000 🡪 0010 1010
12. Exact hetzelfde
13. |=
14. ^=
15. ^=

### Practicum

1. PORTB.DIR &= ~PIN2\_bm;
2. PORTB.IN & PIN2\_bm
3. 1 is indrukken 0 is los
4. PORTB.IN & PIN2\_bm
5. PORTB.IN is de hele bitreeks van de inputwaarden van de port b. PIN3\_bm is een bitmask van pin3 dus als de bit in de bitreeks 1 is, en de juiste bit (de enige hoge bit in de bit mask) hoog is, is de input 1
6. PORTB.DIR |= PIN3\_bm;   
     
   while (1) {  
   if(PORTB.IN & PIN2\_bm)  
    PORTB.OUTSET = PIN3\_bm;  
    } else {  
    PORTB.OUTCLR = PIN3\_bm;  
    }  
   }
7. Ja maar hij blijft aan
9. Zet een pull up resistor aan of uit per pin
10. PORTB.PIN2CTRL, PORT\_PULLUPEN\_bm;
11. PORTB.PIN2CTRL = PORT\_PULLUPEN\_bm;
12. Nu werkt het wel goed
13. Dan wacht je op de input van een extern apparaat inplaats van dat je een trigger (interrupt) krijgt
14. Omdat je niks anders kan toen terwijl je o peen input aan het wachten bent
15. Met een interrupt
16. Timer
17. Gedaan





24. A word elke druk groter










36. int main(void)

{

SYSTEM\_Initialize();

while (1) {

if(SW0\_GetValue()){

LED0\_SetHigh();

} else {

LED0\_SetLow();

}

}

}

2. if(SW0\_GetValue()) { LED0\_SetHigh(); } else { LED0\_SetLow(); };

## 3 Interrupts

### 3.1 Voorbereiding

1. constant bij de deur kijken
2. 1: je staat constant bij de deur dus je kan niks anders  
   2: Het kost enorm veel moeite
3. Je hoeft niet constant bij de deur te staan aangezien je een seintje krijgt als je pakketje er is. Hierdoor kan je iets anders / niks doen om uit te rusten.
4. Polling = geen deurbel, interrups = wel deurbel
5. Er is een externe pheripheral die een seintje stuurt als een bepaalde trigger afgaat
7. GPIO, timer, rtc, event system, adc, dac, uart, spi, i2c
8. PORTB.PIN2CTRL = PORT\_PULLUPEN\_bm;  
   PORTB.PIN2CTRL |= PORT\_ISC\_FALLING\_gc;
9. ISR(PORTB\_PORT\_vect) {

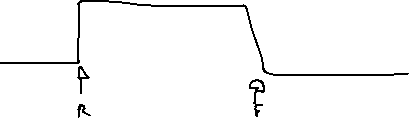
if (PORTB.INTFLAGS & (1 << PIN2\_bp)) {

PORTB.INTFLAGS = (1 << PIN2\_bp);  
 // HIER WORD CODE UITGEVOERD  
 PORTB.OUTTGL = (1 << PIN3\_bp);

}

}

1. Een nested interrupt is een interrupt die onderbroken kan worden door een hogere prioriteit interrupt. Dit is niet altijd maar soms wel wenselijk
2. Andere interrupts moeten wachten tot de isr klaar is dus als het sneller klaar is kunnen de volgende interrups uitgevoerd worden
3. Dat zijn geheugenadressen waar de microcontroller naar toe springt als interrupt getriggered word. Er zijn er 33
4. Rising is wanneer de trigger “ingedrukt”word falling is als hij “losgelaten” word



1. Rising edge trigger aanzetten, falling edge uitzetten
3. Idle, standby, power-down
4. Ja, anders stopt de microcontrooler met de code uitvoeren en crasht / reset hij
5. Idle
6. Pin change interrups,
7. 6

### 3.2 Practicum

8. 0xB
9. 1011
10. ISC en pULLUP enable
11. 1010

14. A screenshot of a computer program

    AI-generated content may be incorrect.
15. #define PORTB\_PORT\_vect \_VECTOR(45) /\* \*/
17. SW0\_InterruptHandler en LED0\_InterruptHandler. Het zijn mijn eigen geconfigureerde pins.
19. A close up of a text

    AI-generated content may be incorrect.je kan dan een bepaalde void doorgeven als argument die dan word uitgevoerd als interrupt handler
20. SW0\_SetInterruptHandler(SW0\_DefaultInterruptHandler);
21. Deze void is leeg


25. 1 = clear en aan het eind van de code worden de flags 0xff gezet dus alles 1
26. Hij start aan en elke klik toggelt hij


30. Hij stopt






38. SLPCTRL.CTRLA en SLPCTRL.VREGCTRL
39. CTRLA
41. Hij start de sleep
42. Werkt nog steeds



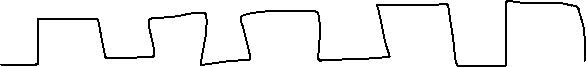


49. A diagram of a code

    AI-generated content may be incorrect.

## 4 Clocks, Timers & PWM

### 4.1 Voorbereiding



1. CLKCTRL
2. OSCHF 1-32MHz  
   OSC32K 32768KHz  
   XOSC32K 32768KHz
3. Die deelt de frequentie om op een lagere frequentie te werken
4. De timing kan vernaggeld worden, de uC kan instabiel worden
5. Een bescherming die het tegenhoud als je kritieke registers wilt aanpassen
6. 100msvraag 181


10. Die vergelijkt constant een waarde met die van de timer en voert vervolgens verschillende dingen uit als die waardes gelijk zijn
11. Met capture kan je op een precies moment een bepaalde waarde vastleggen
12. Compare
13. Capture
14. Compare
15. Compare
16. ???
17. 25%
18. 1000 = x / 20000; x = 1000 \* 20000 = 20000000Hz
19. TOP en CMP waardes aanpassen tot waar het wel kan
20. 6.25%
21. Dimmer of feller
22. Kan de functionaliteit van pins veranderen of pins aan / uit zetten
24. 6
25. 157x groter
26. Dit klopt



31. Hoe hoger de prescaler divider hoe lader de frequentie
33. 4MHz
34. Dat komt door de binare representatie. Het is namelijk 2^15
35. De CPU is enorm traag

38. Die is 4x zo snel als 4MHz
39. Het klopt hij word sneller hoe hoger de frequentie
40. Ja
41. Kost meer stroom

### 4.2 Practicum

4. Hij word dimmer
5. Hij word dimmer
7. HPER, LPER, HCMP0
8. HCMP0

## 5 ADC

### 5.1 Voorbereiding

1. Bij digitaal heb je alleen maar hoog en laag. Bij analoog kan je waarde overal liggen en is heel precies
2. Bij een continu signaal kan je op elk willekeurig tijdstip de exacte waarde verkrijgen. Bij een discreen signaal kan je alleen op bepaalde tijdstippen de (minder) exacte waarde verkrijgen
3. Dat je een gemiddelde waarde pakt
4. Dan kunnen we de gemiddelde waarde van een sample niet berekenen
5. Een event triggered meting
7. 2.048v
8. 0.5mV per stap

|  |  |
| --- | --- |
| Vref | resolutie |
| 1.024 | 0.25mV |
| 2.048 | 0.5mV |
| 4.096 | 1mV |
| 2.500 | 0,6mV |
| 3.3 | 0,8mV |

2. Uit de potmeter komt 0-3.3v
3. ADCn.RES
4. Schakelen tussen inputs
5. Als je 2 inputs wilt meten is het al 2x zo sloom. Dan moet je namelijk omste beurt de 2 inputs meten en kan je minder snel de individuele inputs meten
6. Met een trigger?

### 5.2 practicum