Documentatie Embedded systems Sprint 4

Lucas van Lippen

T3DB

Afstand sensor met PWM input

Code

```
//PWM output op pin PA6 wordt ingesteld met timer 3 en een overflow waarde van 10000 waardoor hij elke 100ms een signaal stuurt met de huidige prescale.
  RCC->APB1ENR |= RCC_APB1ENR_TIM3EN;

  TIM3->PSC = 720-1; //define prescale

  TIM3->ARR = 10000; //define overflow value

  TIM3->CCMR1 = (TIM2->CCMR1 & ~TIM_CCMR1_OC1M) | (0b0110 << TIM_CCMR1_OC1M_Pos);

  TIM3->CCR1 = 1; //pwm write value

  TIM3->CCER |= TIM_CCER_CC1E;

  TIM3->CR1 |= 1;

  GPIOA->MODER = (GPIOA->MODER & ~GPIO_MODER_MODER6) | (0b10 << GPIO_MODER_MODER6_Pos);

  GPIOA->AFR[0] = (GPIOA->AFR[0] & ~GPIO_AFRL_AFRL6) | (0b0010 << GPIO_AFRL_AFRL6_Pos);

```

Hier wordt PWM output ingesteld voor de trigger pin van de afstandssensor. Hij maakt het signaal hoog voor enkel 10us en daarna voor de overige 100ms weer laag. Zoals je ziet is hij ingesteld op pin 6 met timer 3

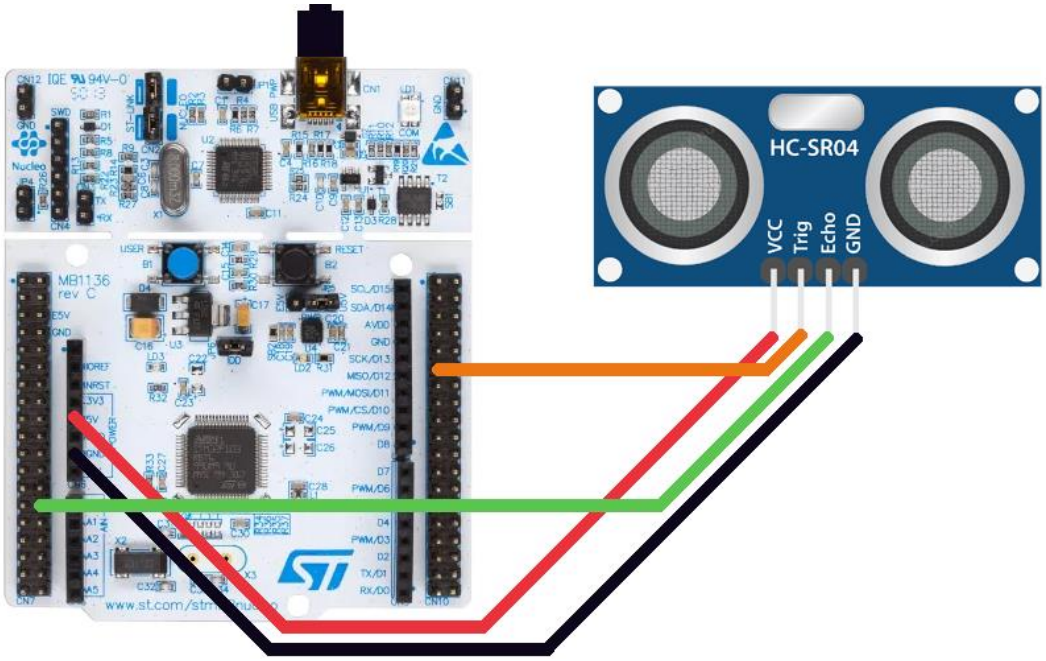
```
//PWM output op pin PA0 wordt ingesteld met timer 2 hij kijkt op de microseconde of de pulse breedte is veranderd we gebruiken timer 2 omdat die ook accurater is.
  RCC->APB1ENR |= RCC_APB1ENR_TIM2EN;  //stap 1

  TIM2->PSC = 72-1; //stap 2

  TIM2->CCMR1 = (TIM2->CCMR1 & ~TIM_CCMR1_CC1S) | (0b01 << TIM_CCMR1_CC1S_Pos); //stap 3

  TIM2->CCER &= ~TIM_CCER_CC1NP;
  TIM2->CCER &= ~TIM_CCER_CC1P; //stap 4

  TIM2->CCER |= TIM_CCER_CC1E; //stap 5

  TIM2->CCMR1 |= (0b10 << TIM_CCMR1_CC2S_Pos); //stap 6

  TIM2->CCER &= ~TIM_CCER_CC2NP;
  TIM2->CCER |= TIM_CCER_CC2P; //stap 7

  TIM2->CCER |= TIM_CCER_CC2E; //stap 8

  TIM2->SMCR |= (0b0101 << TIM_SMCR_TS_Pos); //stap 9

  TIM2->SMCR |= 0b0101; //stap 10

  TIM2->CR1 |= 1; //stap 11

  GPIOA->MODER |= (0b10 << GPIO_MODER_MODER0_Pos);
  GPIOA->AFR[0] = (GPIOA->AFR[0] & ~GPIO_AFRL_AFRL0) | (0b0001 << GPIO_AFRL_AFRL0_Pos); //stap 12

```

Hier wordt de PWM input ingesteld met behulp van timer 2 en gebonden aan pin PA0. Ik gebruik hier een prescaler waarde van 72-1 omdat ik op die manier een puls breedte waarde terug krijg in microseconden omdat het bordje wat ik gebruik 72MHz is.

```
148
149   const int MSGBUFSIZE = 80;
150
151   while (1)
152   {
153
154     char msgBuf[MSGBUFSIZE];
155     double calculationConstant = 4000;
156
157     uint32_t pulseWidth = 0;
158     pulseWidth = (TIM2->CCR2 - TIM2->CCR1);
159
160     double pulseDouble = pulseWidth;
161     double distance = pulseDouble/calculationConstant;
162
163     snprintf(msgBuf, MSGBUFSIZE, "%f", distance );
164     HAL_UART_Transmit(&huart2, (uint8_t *)msgBuf, strlen(msgBuf), HAL_MAX_DELAY);
165
166     snprintf(msgBuf, MSGBUFSIZE, "%s"," cm\r\n" );
167     HAL_UART_Transmit(&huart2, (uint8_t *)msgBuf, strlen(msgBuf), HAL_MAX_DELAY);
168
169     HAL_Delay(200);
170   }
171 }
```

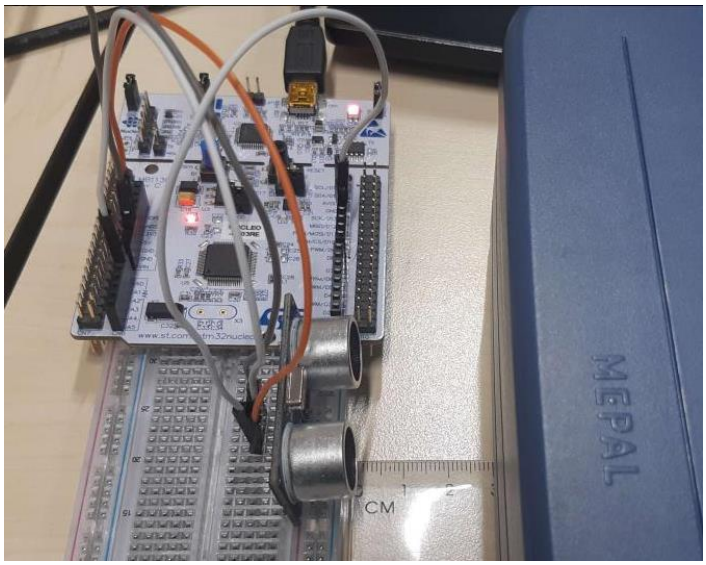
Hier wordt de puls breedte omgezet naar een afstandswaarde in centimeters, deze waarde wordt daarna ook gelijk uitgeprint.

Opstelling

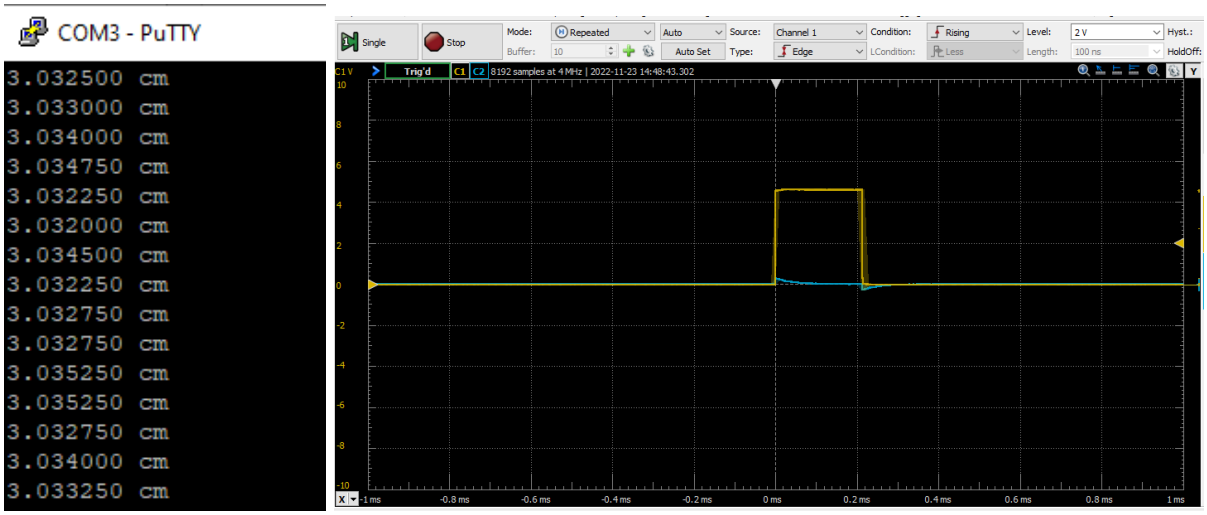


Resultaten

3 centimeter

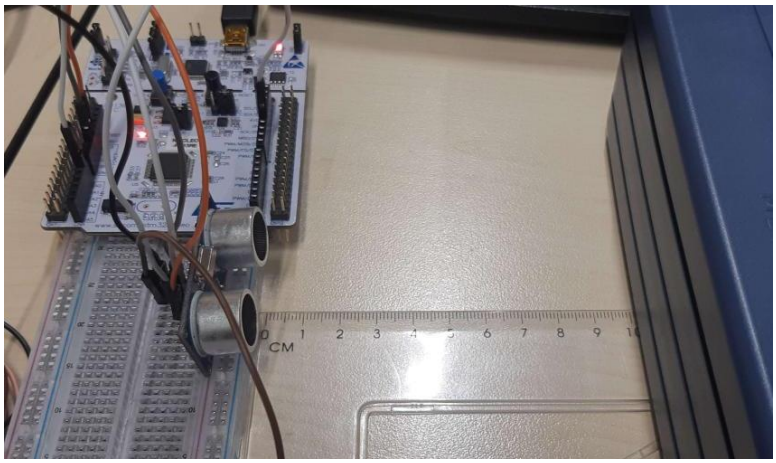


De afstand van de afstand sensor tot de doos.

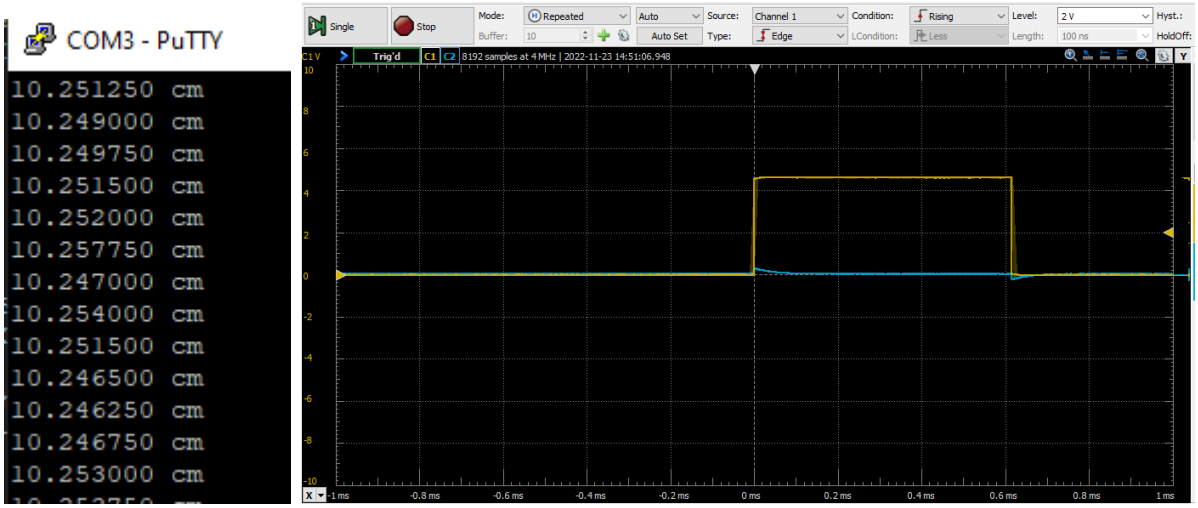


Hier kun je zien dat hij een afstand van 3 centimeter laat zien en dat de puls breedte 0,2 ms breed is.

10 centimeter

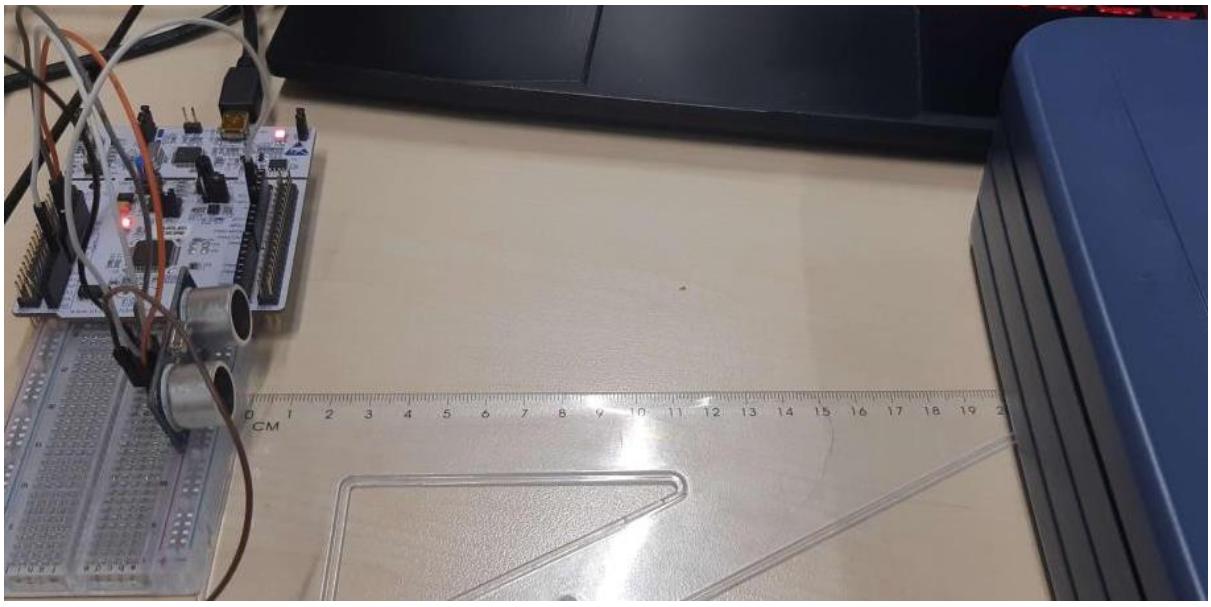


De afstand van de afstand sensor tot de doos.

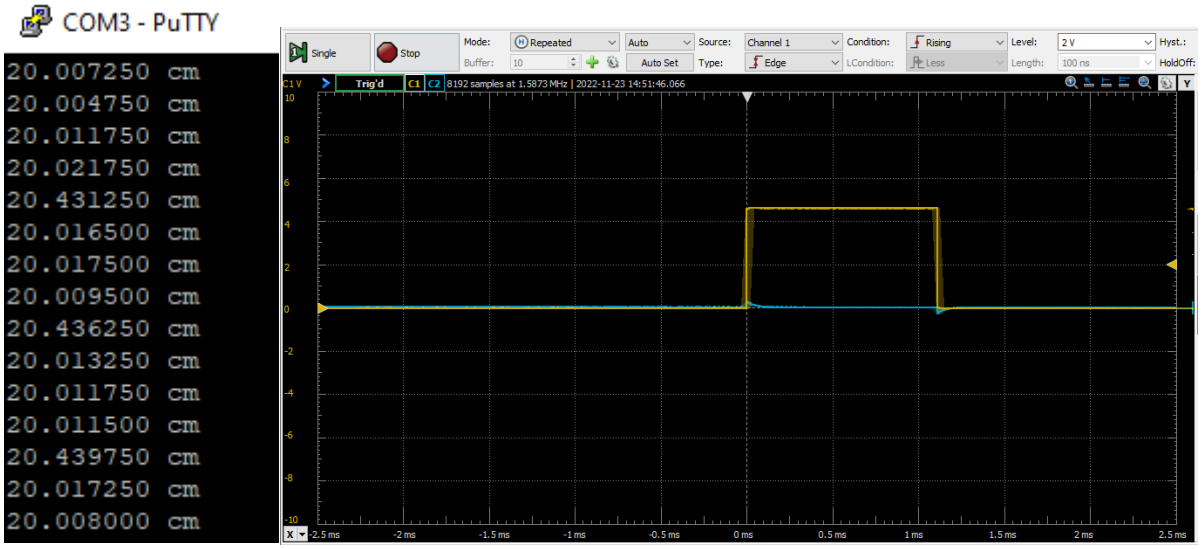


Hier kun je zien dat hij een afstand van 10 centimeter laat zien en dat de puls breedte 0,6 ms breed is.

20 centimeter

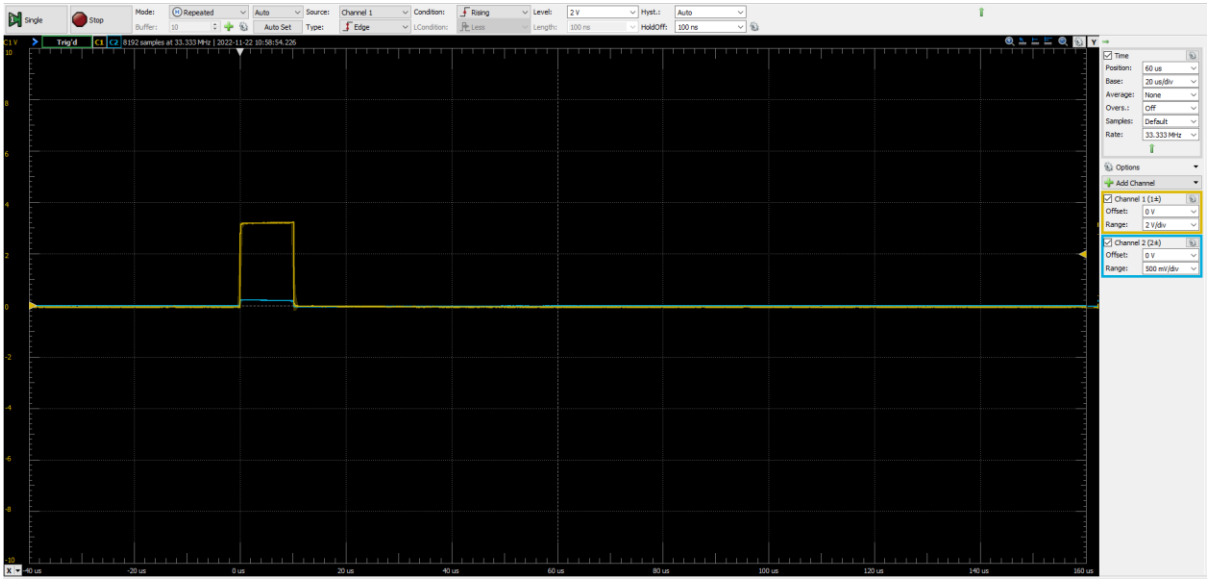


De afstand van de afstand sensor tot de doos.

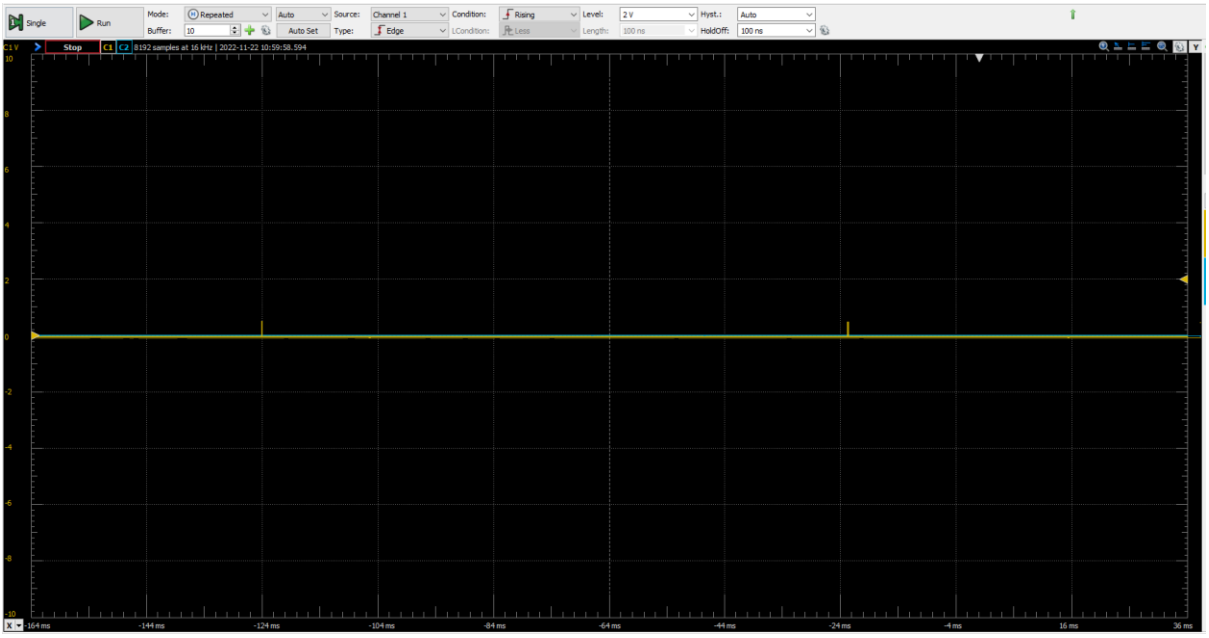


Hier kun je zien dat hij een afstand van 10 centimeter laat zien en dat de puls breedte 1.2 ms breed is.

Output puls signaal



De trigger puls zoals je die hier op de usb oscilloscoop ziet heeft een breedte van 10 micro seconden.



Hij wordt ook elke 100ms ongeveer aangeroepen.