ES ASSIGNMENT 3

Knight Rider

Thijs van de Weijer
Rens Pastoor
Gijs van Maanen

Inhoud

Doelstelling

Het doel van deze opdracht was om registers (DDRx, PORTx, PINx) en bit manipulatie toe te passen voor directe aansturing van digitale I/O op de Arduino Uno. De opdracht bestond uit twee delen:

Opdracht A: Interne LED aan zetten door middel van een button.

Opdracht B: Knight Rider animatie met één of meerdere LEDs, afhankelijk van een tweede knop

Hardware

1x Arduino UNO

2x Button

5x LED

5x 470Ω weerstanden

Jumper wires

Werking van de Software

Opdracht A – Interne LED Blink

In opdracht A ga je aan de slag met twee LED's en een knop, maar dan zonder de standaard Arduino-functies zoals digitalRead of digitalWrite. Je gebruikt hiervoor direct de registers van de microcontroller, wat wat laag-niveau programmeren is.

Het idee is dat een externe LED op pin 7 continu knippert met een snelheid van 4 keer per seconde. Dat doe je door zelf bij te houden hoe laat het is met millis() en telkens na 125 milliseconden de LED aan of uit te zetten. Zo voorkom je dat het programma vastloopt, want je gebruikt geen delay().

Daarnaast is er een knop op pin 5, die je ook handmatig uit moet lezen. Zodra je die knop indrukt, gaat de ingebouwde LED van de Arduino (die op pin 13 zit) aan. Omdat knoppen vaak ruis geven als je ze indrukt (de knop 'stuitert'), moet je dit netjes oplossen met een soort pauze tussen het lezen van de knop, ook wel debouncing genoemd. Dit doe je weer met millis() om snel meerdere leestijden te vermijden.

Zo heb je een systeem waarbij de externe LED lekker knippert, en de ingebouwde LED precies aan gaat als je de knop indrukt, en dat alles tegelijk werkt

Code snippet(built in led):

```
if (button1State) {
    PORTB |= LED_BUILTIN_MASK;
} else {
    PORTB &= ~LED_BUILTIN_MASK;
}
```

Code snippet(debounce button 1 (button 2 is the same)):

```
bool reading1 = (PIND & BUTTON1_MASK);
if (reading1 != lastButton1Reading) {
    prevMillisDebounce1 = currentMillis;
    lastButton1Reading = reading1;
}
if (currentMillis - prevMillisDebounce1 > 20) {
    button1State = reading1;
}
```

Opdracht B – Knight Rider (Sweep A & B)

In opdracht B voeg je vier extra LEDs toe die je op pins 8 tot 11 aansluit, plus een tweede knop op pin 2. Met die LEDs maak je een “Knight Rider”-effect: een lichtje dat heen en weer beweegt.

Er zijn twee varianten van dit effect, afhankelijk van de status van die tweede knop. Als de knop **niet** ingedrukt is, beweegt er steeds één LED van links naar rechts en weer terug — dat noemen we Sweep A. Maar zodra je de knop indrukt, verandert het patroon naar Sweep B: hierbij gaan eerst één LED aan, daarna twee, daarna drie, tot alle vier aan staan, en daarna dimmen ze weer af, steeds in die volgorde.

De code leest de knop uit met debouncing, zodat het schakelen soepel verloopt. Ook regelt de code met millis() wanneer de LEDs van status wisselen, zodat alles netjes op tijd gebeurt zonder vertraging.

Het fijne is dat deze animatie tegelijk kan draaien met de knipperende LED uit opdracht A en de ingebouwde LED die reageert op de eerste knop, allemaal zonder dat ze elkaar in de weg zitten. Zo leer je hoe je meerdere taken tegelijk kunt laten lopen op de Arduino, door slim gebruik van registers en timing

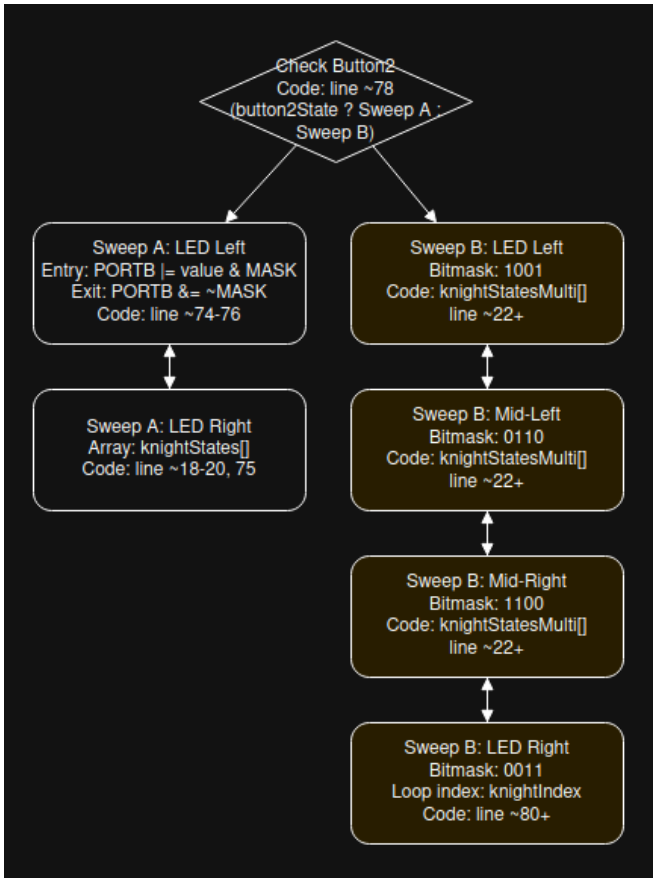
Gebruikte technieken

- Bit manipulatie: |=, &= ~, << om individuele bits aan of uit te zetten.
- Registers: DDRx, PORTx, PINx gebruikt voor alle I/O, conform opdrachtvereisten.
- Tijdmeting met millis() om knipperfrequentie en debouncing te regelen.
- Efficiënte LED-sturing: met behulp van bitmask arrays (knightStates, knightStatesMulti).

Code snippet (bitmask):

```
uint8_t knightStatesMulti[] = {
    0b0001,  // 1
    0b0011,  // 12
    0b0111,  // 123
    0b1111,  // 1234
    0b1110,  // 234
    0b1100,  // 34
    0b1000,  // 4
    0b1100,  // 34
    0b1110,  // 234
    0b1111,  // 1234
    0b0111,  // 123
    0b0011   // 12
};
```

State diagram:



Resultaat

De implementatie werkt zoals bedoeld:

- Externe LED knippert 4x per seconde.
- Ingebouwde LED reageert op knop A.
- Knight Rider LED's tonen twee verschillende sweeps afhankelijk van knop B
- Sweep A: enkele LED beweegt heen/weer.
- Sweep B: meerdere LEDs bouwen op tot alle 4 aan, en daarna weer af.
- Alle functies werken gelijktijdig dankzij tijdgebaseerde logica.

Conclusie

De opdracht is succesvol uitgevoerd met volledige toepassing van bit manipulatie en registerprogrammering. Beide delen (A & B) functioneren onafhankelijk en parallel. De code is modulair opgebouwd, goed gedocumenteerd en voldoet aan de opdrachtcriteria zoals beschreven in het practicumdocument.