Practicum

Track 6



**Cas Koopmans**

**09/03/2017**

# Repository

Deze repository bevat de volledige code die voor deze opdrachten gemaakt zijn.

# <https://github.com/CasKoopmans/MicrocontrollersAssignments>

# Opzoekvragen

1. Hoe heet de adresseermode in: *ldi yh,$04*

*Data Indirect Adressing*

1. Wat is het verschil tussen: *mov r1, r2* en *ld r1,6*

*Bij mov wordt het ene register naar de andere gekopieerd, bij ld wordt een waarde in het register geschreven.*

1. Hoeveel klokpulsen zijn nodig voor: adc r5,r7

*1 klokpuls*

1. Hoeveel verschillende ‘load’-instructies zijn er?

*22 load-instructies in de AVR instruction set*

1. Schrijf een instructie/ instructies om een geheugenplaats waarvan het adres in register *r12* zit een waarde 88d te geven.

*ldi r12, 58*

# Coding

Laat het programma voor de optellen van 16-bits getallen draaien; de code staat in bijlage 6.1

(en op Blackboard, dag6\_b):

1. Laad de code, start de debugger, en bekijk het code en het data geheugen, ga stap voor stap door code heen (F11).

*We zijn door de debugger heen gelopen, en hebben de veranderende waarden in het programmageheugen bekeken.*

1. Uit hoeveel bytes bestaat de code van dit programma, hoeveel cycles zijn nodig voor de executie? Geef een tabel met per regel een instructie en

#bytes en #cycles en het totaal.

.ORG $0200 | 200 bytes | 1 cycle | 200 totaal

LDI YH, $04 ; Y <- $0400 adres in data memory | 1 byte | 1 cycle | 201 totaal

LDI YL, $00 ; | 1 byte | 1 cycle | 202 totaal

LDD R8, Y+0 ; Haal waarden op uit data memory | 1 byte | 2 cycles | 203 totaal

LDD R9, Y+1 ; | 1 byte | 2 cycles | 204 totaal

LDD R6, Y+2 ; | 1 byte | 2 cycles | 205 totaal

MOV R4, R8 ; R4 <- R8 | 1 byte | 2 cycles | 206 toaal

ADD R4, R6 ; R4 <- R4+R6 misschien Carry gezet | 1 byte | 1 cycle | 207 totaal

MOV R5, R9 ; R5 <- R9 | 1 byte | 1 cycle | 209 totaal

ADC R5, R7 ; R5 <- R5+R7+C | 1 byte | 1 cycle | 20A total

STD Y+4,R4 ; Sla resultaat op in data memory | 1 byte | 1 cycle | 20B total

STD Y+5,R5 ;

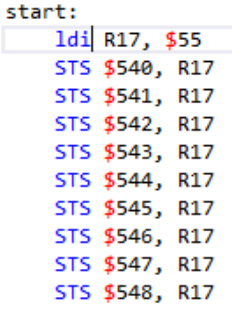
no\_end: ; unlimited loop, when done | 1 byte | 2 cycles | 20C totaal

1. View code geheugen: de eerste 10 bytes en de bytes vanaf 0200H, klopt het met b) ?
2. Zet een breakpoint aan het einde, laat het programma runnen in de debugger (‘go’), klopt het aantal cycles met b)

*C en D zijn bij opdracht a beantwoord.*

# Coding vervolg

1. Schrijf een stukje assembler programma om de geheugenplaatsen in het datageheugen met adres 540H tot en met 548H de waarde 55H te geven



*We zetten de waarde van register 17 op 55 hex. Dit kopiëren we vervolgens naar adressen 540-548.*

1. Maak een (assembler) programma dat het 1-complement bepaalt van degeheugenplaatsen in het datageheugen tussen adres 0a60H en 0a70H. (Er is een instructie waarmee je het one’s complement kan bepalen).

start:

lds R17, $60

com R17

STS $60, R17

end:

rjmp end

1. Maak een (assembler) programma dat waarden van de geheugenplaatsen in het datageheugen van adres 660H t/m 670H kopieert naar de geheugenplaatsen 8B0 t/m 8C0H.

(Zie het voorbeeld uit de les, dag6\_d)

start:

lds R16, 10

ldi xh, $66

ldi xl, 00

ldi yh, $8B

ldi yl, 00

loop:

ld R17, X+

ST Y+, R17

dec R16

brne loop

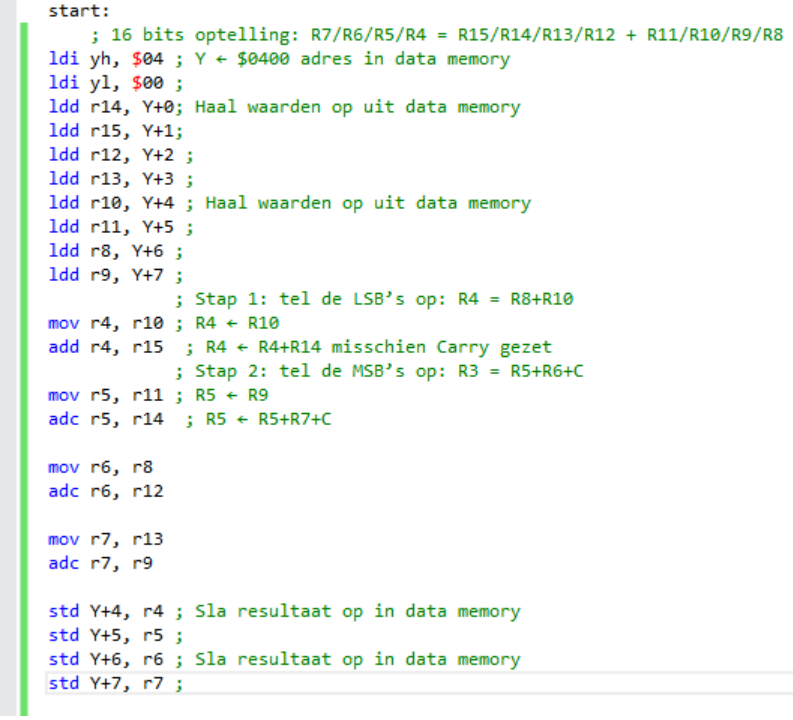
end:

rjmp end

1. Bepaal uit hoeveel geheugenplaatsen het programma van 6.5 bestaat en hoeveel klokpulsen dit programma in beslag neemt. Laat een berekening zien in een duidelijke tabel.

*Het programma kost 128 clock cycles, en heeft 210 bytes aan program memory.*

1. Maak een (assembler) programma dat een 32-bits optelling van twee 32-bits getallen doet (Add32, dit is een uitbreiding van Add16 met telkens 4 bytes).



1. Maak een programma RunLight voor de LED’s van PORTA. Laat de LED’s heen en weer lopen (van 0 🡪 7 en weer van 7 🡪 0).

start:

ldi R16, 0

ldi R17, 7

out PORTA, R16

loop:

inc R16

out PORTA, R16

dec R17

brne loop

back:

ldi R17, 7

secondloop:

dec R16

out PORTA, R16

dec R17

brne secondloop

end:

rjmp start