Indhold

[Opgaven 1](#_Toc523760127)

[Register oversigt 1](#_Toc523760128)

[ICO diagram 1](#_Toc523760129)

[Flowchart 2](#_Toc523760130)

[Hjælp - Instruktioner 3](#_Toc523760131)

[Eksklusive OR --- X-OR 3](#_Toc523760132)

[Eller/OR - gate 3](#_Toc523760133)

[Og/AND - gate 3](#_Toc523760134)

[Check om to tal er ens 3](#_Toc523760135)

[Deffinition af registre 4](#_Toc523760136)

[Deffinition af konstanter 4](#_Toc523760137)

[Bedømmelse 4](#_Toc523760138)

# Opgaven

Denne opgave henter viden fra de tidligere labøvelser 2,3 og 4. I skal lave en kodelås der, når i indtaster 4 kombinationer af 8 bit låser en dør op.

For at det kan lade sig gøre at taste fire kombinationer, er det nødvendigt at gemme alle de taster der har været tastet, i et register. Det må forventes at tasterne aktiveres en af gangen, og at de ligeledes slippes en af gangen. [Se hvad der bedømmes på.](#_Bedømmelse)

1. Taste kombinationen der giver succes, skal skrives i kommentar feltet i programmets top.
	1. Her vil det være smart at deffinerer 4 [konstanter](#_Konstanter)
2. Når der tastes et tal skal ”LED 5” lyse. ( indikerer bib )
3. Når der tastes forkert skal ”LED 6” lyse. ( indikerer fejl lyd efter 4.de kombination, Debugging)
4. Tastes 4 rigtige skal LED 7 lyse i 5 sek. ( låsen låser op )

# Plan

1. Start med at aflæse 8 bit, så alle bit der har været aktiveret huskes
2. Sammenlign de 8bit med et defineret mønster tænd LED7 (ok) eller LED6 (fejl)
3. læg aflæsning og sammenligning i en procedure (Call og return)
4. Aflæs 4 koder
5. Få LED´s til at virke (uden tid)
6. sæt tid på led 7

# Register oversigt

Der skal laves en register oversigt over de registre i bruger og evt. hvilke bits der anvendes til hvad.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Registernavn  | Adr. | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit1 | Bit 0 |
| Kombiantion1 | 0x20 |  |  |  |  |  |  |  |  |
| Kombination2 | 0X21 |  |  |  |  |  |  |  |  |
| Resten komme i på  | 0X22 |  |  |  |  |  |  |  |  |

# ICO diagram

Der skal laves en oversigt over hvad der er fungerer som indgange, hvad der behandler de indkommende data og hvad skal være af udgange. ICO diagrammet er et blokdiagram hvor hver enhed får sin egen blok, i dette tilfælde er alle vores switches samlet i et tastatur, og vores LEDs er separate indikatorer.

Binært tastatur

PIC16F877a

Indgange

Controler

Kode OK - LED

Udgange

Sæt selv de sidste udgange på.

# Flowchart

Der skal som minimum laves et overordnet flowchart, der beskriver den grove funktion af programmet.
I elteknik laver vi flowcharts af tre bokse, Start, process og spørge boks med 2 udgange.

Start/ slut

Spørgsmål

Ja

Nej

Process

# FAQ

1. Hvordan tjekker man, om det som er gemt i w, passer med den værdi der står i Registret ens kode (ens)?

 SUBWF Registeret,W ; W trækkes fra Registeret og resultatet lægges i W

 BTFSS STATUS,Z ; Hvis resultatet blev 0=Zero springes næste instruktion over

 Goto HvisUEns ; spring til proceduren uens hvis W og registeret er ens

 Goto HvisEns ; spring til proceduren uens hvis W og registeret er ens

1. Hvordan kan man forklare kodningen og MPLab noget bedre så elever uden erfaring med programmering kan forstå bedre?
Rigtig godt spørgsmål, Jeg kæmper med det hvert år, Den første del af problemet er at i må give jer til kende. I må spørge, og spørge igen. Jeg kan sagtens se der er nogen der er hurtigere end andre, men i 2018 er vi nået langt på de første 2 uger. Så der er tid til at hjælpe dem, der føler det er svært. Men vi vil ikke putte hjælpen ned i halsen på jer. Så når vi kommer forbi må i sige dette er hvad jeg har nået, hvordan kommer jeg videre. Jeg glæder mig til at komme og hjælpe.
2. Det kunne være rart og praktisk, hvis vi kunne få en slags liste over kommandoer og funktioner
I alle zipfilerne til labøvelserne ligger 2 filer ” Instruction set den store model.pdf” og den mindre ” Instruktionssæt.xls” der har alle 35 instruktioner på en side.
Når der ikke er lagt instruktioner ind på denne side er det fordi, problemstillingen kan løses på mange måder, og de vil kræve forskellige instruktioner.
3. Hvorfor skal Led6 lyse efter fejl indtastning
Led 6 indikerer at der er foretaget 4 indtastninger, og at resultatet ikke giver adgang til området
4. Tilføjelser for de hurtige:
Identifikation af hvem af 6 personer, med forskellig kode, der er i området, 8 bit til identifikation 4\*8bit til koden

# Hjælp - Instruktioner

Eksklusive OR --- X-OR
Der bliver behov for at teste en bit i et register, her er det muligt at anvende en X-or funktion på enkelte bits
Sandhedstabel X-or set på et helt register

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A register | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| B register  | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| Q Resultat | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |

X-Or gate

|  |  |  |
| --- | --- | --- |
| 1.Indgang | 2.Indgang | Udgang |
| A | B | Q |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

## Eller/OR - gate

På en ”OR” gate, skal enten den ene indgang, den anden indgang eller begge indgange være 1 for at udgangen bliver 1.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A register | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| B register  | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| Q Resultat | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |

OR gate

|  |  |  |
| --- | --- | --- |
| 1.Indgang | 2.Indgang | Udgang |
| A | B | Q |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

## Og/AND - gate

På en ”AND” gate skal begge indgange være 1 for at udgangen er 1

Ud over logiske operationer på bits er der en del instruktioner det kunne være godt at kende til.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A register | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| B register  | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| Q Resultat | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |

AND gate

|  |  |  |
| --- | --- | --- |
| 1.Indgang | 2.Indgang | Udgang |
| A | B | Q |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

## Check om to tal er ens

Hvis vi trækker to tal fra hinanden, kan vi få et positivt tal så giver det ikke meget information.
Det giver mere information hvis resultatet giver nul, så er de to tal lige store, og nu sættes Zero bit´en se nedenfor.
Hvis vi får et negativt resultat vil Carry bit´ten gå hen og blive nul.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Registernavn | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2  | Bit1 | Bit0 |
| Status  | 0 | 1 | 0 | 0 | 0 | Zero | Decimal carry | Carry |

**Bit test File skip if set** -- **BSFSS** kender vi, men vi kender endnu ikke **STATUS** registeret, det indeholder nogle bits der kan være ganske nyttige

Den følgende kommando springer over den næste hvis den forrige kommando gav et nul
BTFSS Status, Z

## Deffinition af registre

Når vi definerer registre, skal de i reglen ligge på adresserne 0x20 til 0x7F
TempKode equ 0x20

## Deffinition af konstanter

konstanter defineres på samme måde, som registre
kode1 equ b’00101000’

# Bedømmelse

Vi sigter altid på eksamen. Så der lægges stor vægt på dokumentationen.

|  |  |
| --- | --- |
| Krav beskrivelse  | point |
| ICO diagram | 0-10 |
| Flowchart | 0-10 |
| Register oversigt  | 0-10 |
| Kommentar hoved først i programet | 0-10 |
| Afsnit kommentarer  | 0-10 |
| Linje kommentarer | 0-10 |
| Test af kode1 | 0-10 |
| Test af kode 1-4 | 0-10 |
| Aktivering af indikations LED´s | 0-10 |
| Sikkerhed ved fejl indtastning | 0-10 |