

Elektroteknik MF1016 och MF1017 föreläsning 7 (9) Mikrodatorteknik

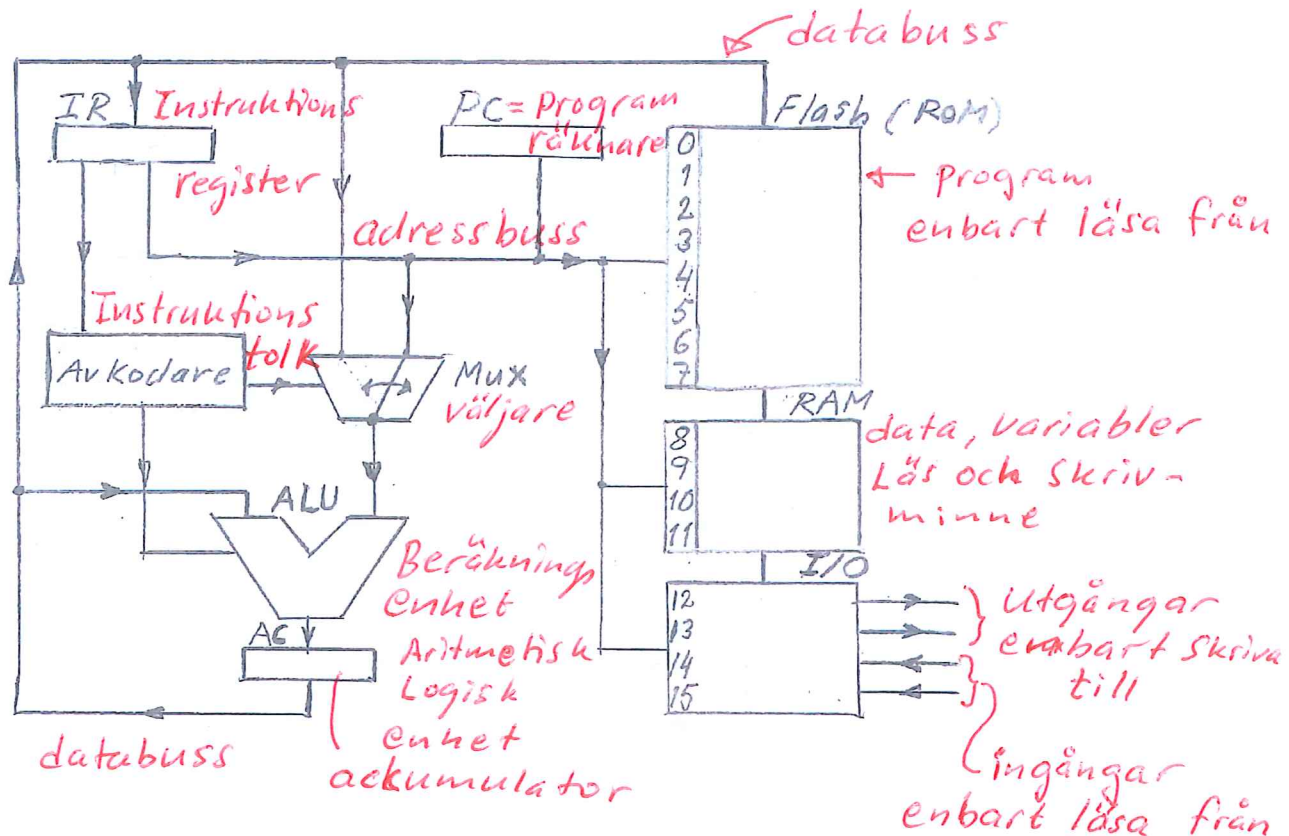
- **Inbyggda system**
- **Uppbyggnad av mikrokontroller**
- **Masinkod, assemblerkod**
- **Högnivåkod**
- **Utvecklingsystem, målsystem**

Datorer och programvara i produkter som inte i sig är datorer kallas inbyggda system (embedded systems).

Inbyggda system måste ibland vara säkerhetskritiska. Man talar om säkerhetskritiska system.

http://news.bbc.co.uk/2/hi/programmes/click_online/9755210.stm

<http://courses.cs.vt.edu/~csonline/MachineArchitecture/Lessons/CPU/Lesson.html>



Hämtfas: Instruktion hämtas ur ROM och läggs i IR

Utförandefas: Instruktionen avkodas till kontrollsignaler och utförs

LOAD #2 // laddar talet 2 till AC
 STORE 8 // lagrar innehållet i AC till minne 8 i detta fall (2).
 LOAD #5 // 5 → AC
 STORE 9 // AC → minnescell 9 (5)
 LOAD 8 // Laddar innehållet i minne 8 till AC (2)
 ADD 9 // Adderar innehållet i AC (2) och innehållet i minnescell 9 (5) och lägger summan i AC (7)
 STORE 10 // AC → minne 10 (7)

Elektroteknik MF1016 och MF1017 föreläsning 7 (9) Mikrodatorteknik

Här kan du köra ett C program för att testa: <http://codepad.org>
Du kan till exempel klippa ut nedanstående C-program och exekvera det.

```
#include <stdio.h>
//deklarationer och initieringar
int a=1, b=1, c, t=20;
int main(void)
{
    //Här börja tidsnedräkningen
    while (t>0)
    {
        if ((a==1)&&(b==1))c=1;    // Beräkning av utsignal
        else c=0;                //i detta fall AND-funktion

        printf("tid kvar=%i c=%i \n",t,c); //skriv utsignal
        //tiden som är kvar minskas
        t=t-1;
    }//while slutar här

    return 0;
} //main slutar här
```

Nedanstående program kan inte köras i codepad.
Det kan däremot köras på vår mikrokontroller i labbet.

grind.c

```
// grind.c, ett program för MET-kontrollern
// Långlab Mikro, Kurs Elektroteknik
// MG & TL 2002
// Modifierad för ATmega128 PK 2009 och 2012 HJ

#include "gnu_met3.h" // Infogar bibliotek medfunktioner

char * prog = "Grind"; // Textsträng med programmets namn
int ver = 0; // ÖKA numret för varje version ni
// flashar! ***

int a, b, c; // Deklaration av variablerna

int main(void) // Själva programslingan
{
    init_met(); // Initierar MET-kontrollern

    move_cursor(1,1); // Displaymarkören till rad 1,
    //kolumn 1
    dprintf("%s v.%i", prog, ver); // Skriv ut programmets
    //namn/version

    init_pe(0,"in"); // Initierar pe.0 som inpinne
    init_pe(1,"in"); // Initierar pe.1 som inpinne
    init_pe(2,"out"); // Initierar pe.2 som utpinne

    while(1) // Evighetsloop!
    {
        a = GET_BIT(pe,0); // Läser av pe.0
        b = GET_BIT(pe,1); // Läser av pe.1
        if ((a==1)&&(b==1))c=1; // Beräkning av utsignal
        else c=0; //i detta fall AND-funktion
        if(c == 1) SET_BIT(pe,2); // Utsignal beroende på c
        else CLR_BIT(pe,2);

        // Programslingans slut
    }
}
```

*Om och
Om igen.
Cykl, cykl,
...*

*"Läser" insignaler
Beräknar
utsignaler
"skriver"
utsignaler*

*Alltid lika "nästan" skriver ut
programnamn och version på display*

2012-10-02/HJ

*specifikt för hårdvaran ifråga, ej standard
C, pinnarna är bidirektionella.
De kan antingen vara ingångar eller
utgångar. Här ger vi direktiv om
hur vi vill ha det.*