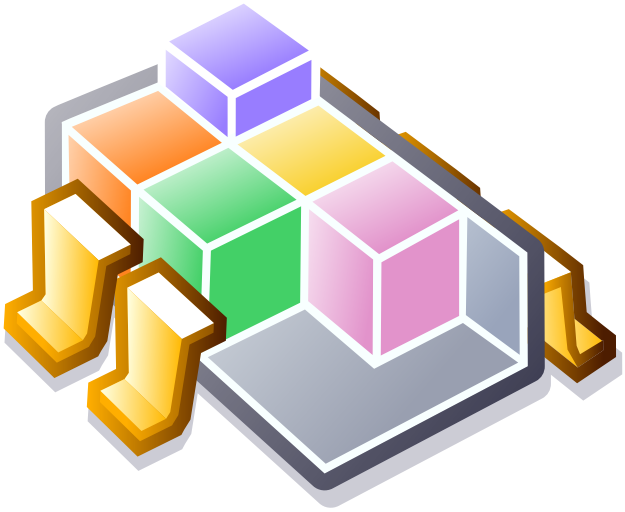
Handbok KTechlab

David Saxton
Daniel Clarke



Innehåll

Sammanfattning

Ktechlab är en integrerad utvecklingsmiljö för mikrokontroller och elektronik.

Kapitel 1

Snabbtur

1.1 Inledning

KTechlab är en integrerad utvecklingsmiljö för mikrokontroller och elektronik. Den kan utföra simulering av ett antal komponenter (logisk, integrerad, linjär, icke-linjär och reaktiv), simulering och avlusning av PIC mikrokontroller via gpsim, och levereras med sina egna tätt kopplade och komplementära högnivåspråk, FlowCode och Microbe.

Den har konstruerats för att vara så lättanvänd och lite påträngande som möjligt. Alla komponenter och flödeselement har sammanhangsberoende hjälp, och simulera elektronik är så enkelt som att dra komponenter till arbetsområdet och skapa kontakter som automatiskt kopplar ihop sig via sina anslutningar. FlowCode låter nya användare av PIC:er omedelbart skapa egna program, medan elektroniksimuleringen gör det möjligt att stega igenom ett PIC-assemblerprogram inne i en krets.

1.2 Dokument

För att komma igång med detta program, måste du skapa ett nytt dokument, där typen beror på din avsikt:

- FlowCode-dokument: Skapa ett PIC-program via ett flödesschema.
- Kretsdokument: Simulera elektronikkretsar och mikrokontroller.
- Microbe-dokument: Högnivåspråk för PIC:er, också använt av FlowCode för att skapa assemblerkod.
- Assemblerdokument: Börja skriva ett PIC-assemblerprogram.

KTechlab använder en dokumentvisningsmodell, på så sätt att dokumentets logik är helt skild från öppna vyer av dokumentet. Det tillåter flera vyer av samma fil.

När ett nytt dokument skapas, skapas vyn under en separat flik. Varje flik kan stödja hur många vyer som helst, ordnade i ett godtyckligt mönster. Det möjliggör exempelvis simulering av ett PIC-program i en krets, medan programmet stegas igenom i ett assemblerdokument under samma flik.

Innehållet i flikar kan dupliceras genom att dra fliken till ett tomt område på flikraden. De kan infogas i en befintlig flik genom att dra dem till den fliken.

Detaljerade instruktioner om dokumenten ovan finns i de respektive egna kapitlen.

1.3 Rita

Det finns flera ritverktyg inklusive text tillgängliga i krets- och FlowCode-dokument. De är tillgängliga genom att klicka på penselikonen i verktygsraden. Dra musen för att antingen skapa en form eller en linje lämplig för ritverktyget som används.

När en ritning är markerad kan dess storlek ändras genom att dra i greppen. Att hålla nere **Skift** under dragning, låser greppen till det underliggande rutnätet. Varje verktyg har grundalternativ tillgängliga i verktygsraden, såsom färger. Det finns också mer avancerade alternativ i sidoraden **Objekteditor**, såsom linje- och ändstilar.

Kapitel 2

PIC-program

2.1 Hantering

När ett FlowCode- eller textdokument skapas, visas en kombinationsmeny i verktygsraden med en raketikon. Därifrån kan man hantera PIC-programmet, och ändra det till olika former.

- **Konvertera till Microbe**: Det används bara i FlowCode-dokument, och förklaras ytterligare i kapitel 4.
- **Konvertera till assembler**: Det kan användas i fyra sammanhang. När ett FlowCode-dokument är öppet, skriver det ut FlowCode som assemblerinstruktioner. När ett Microbe-dokument är öppet kör det programmet **microbe** som distribueras med KTechlab för att kompilera programmet. På liknande sätt, om ett C-program är öppet, försöker det kompilera det via SDCC. När ett textdokument som innehåller PIC-hexkod är öppet, startas **gpdasm** för att disassemblera hexkoden.
- **Konvertera till hexkod**: Det kan också användas i fyra sammanhang. Liksom med konvertera till assembler kan det användas med FlowCode, Microbe och C-dokument. Det aktiveras också när ett assemblerdokument är öppet för att assemblera det via **gpasm**.
- **Ladda upp till PIC**: Det assemblerar PIC-programmet som för närvarande redigeras, och laddar upp det genom att använda programmeraren som användaren har valt.

Ingen av åtgärderna kräver att det aktuella dokumentet sparas, mycket användbart när ett snabbt program behövs. För andra mål än PIC, kan dialogrutan **Utmatning**, som visas när någon av åtgärderna väljes, antingen mata ut resultatet (alltid text i de tre ovanstående fallen) i ett nytt dokument eller till en fil. Om utdata sparas i en fil, ger den också alternativ för att läsa in filen efter den skapats och lägga till den nyskapade filen i det öppna projektet (om något är öppet).

Observera att det går att få KTechlab att alltid använda samma vy för att visa det utmatade innehållet genom att välja alternativet under **Allmänna inställningar**.

2.2 Uppladdning

KTechlab använder tredje-part-programmerare för att ladda upp program till PIC:er. En mängd vanliga programmerare är fördefinierade, medan andra kan läggas till via dialogrutan **Inställningar**.

Portlistan erhålls från avsökning av seriella och parallella portar som är läs- och skrivbara. Seriella portar söks efter bland:

- /dev/ttyS*[0..7]*
- /dev/tts/*[0..7]*
- /dev/ttyUSB*[0..7]*
- /dev/usb/tts/*[0..7]*

Parallella portar söks efter bland:

- /dev/usb/parport*[0..7]*
- /dev/usb/parports/*[0..7]*

Kapitel 3

Kretsar

3.1 Placera komponenter

Till vänster hittar du fliken **Komponenter**.

Att dra en komponent från sidoraden till kretsen placerar den under muspekaren. Som alternativ kan man dubbelklicka på ett objekt i sidoraden **Komponenter** för att lägga till den i kretsen upprepade gånger. Med den metoden placeras en kopia av den markerade komponenten upprepade gånger vid vänsterklick med musen, ända tills ett tryck på Escape eller högerklick med musen.

För att flytta på en komponent, vänsterklicka och dra. Du kommer att märka att den låser till det underliggande rutnätet. Om komponenten dras utanför arbetsområdets höger- eller underkant, ändrar arbetsområdet storlek för att anpassa sig.

Alla komponenter har ett begrepp om orientering: 0, 90, 180 och 270 grader. De är inte symmetriska runt en axel, utan kan också vändas. För att rotera ett antal markerade komponenter, högerklicka antingen och välj i menyn **Orientering**, eller klicka på rotationsknapparna i verktygsraden. De senare kan också kommas åt genom att trycka på tangenterna **[** och **]** (välkänt för användare av Inkscape). Sidoraden **Objekt** (till höger) tillhandahåller en effektiv metod för att ställa in orienteringen genom att visa förhandsbilder av komponenterna. Att vända komponenter är också möjligt via sidoraden **Objekt**.

3.2 Koppla ihop komponenter

Det finns två sätt att skapa kopplingar (ledningar): automatiskt och manuellt. Sätten väljes via kombinationsmenyn **Anslutningsmetod** i verktygsraden. Experimentera med båda. Automatisk koppling är oftast bättre för små kretsar, medan mer komplexa kretsar kan behöva manuell koppling.

Med den automatiska metoden skapas en koppling genom att antingen dra från en komponentanslutning eller befintlig koppling, och släppa musen över önskad anslutning eller koppling. Den raka linjen som dras blir orange när en giltig koppling skulle skapas när musen släpps. Om linjen som ritas är svart beror det antingen på att det inte finns någonting under muspekaren, eller att du försöker koppla ihop två objekt som redan är hopkopplade. Med flödesscheman är kriterierna för en giltig koppling komplexare, men vi kommer till det senare.

Det bästa sättet att få en känsla för manuell koppling är att experimentera med den. Klicka på startanslutningen eller kopplingen, och sträck ut den nya kopplingen genom att flytta musen bort från stället som klickades. Vänsterklicka för att placera ett hörn. Tryck antingen på Escape eller högerklicka med musen för att avbryta uppritning av kopplingen.

KTechlab försöker så gott som möjligt behålla vägarna som dina kopplingar går. Om att dra en komponent dock gör att en kopplings slutpunkter rör sig i förhållande till varandra, tvingas KTechlab att rita om kopplingen med den automatiska metoden. Innan en komponent flyttas kan du se vilka kopplingar som måste dras om, eftersom de blir gråa när de klickas.

För att ta bort en befintlig koppling, markera den genom att rita upp en liten markeringsrektangel över en del av kopplingen, och tryck på **Delete**.

3.3 Komponentegenskaper

De flesta komponenter har redigerbara egenskaper, såsom motståndsvärdet för motstånd. Normalt kan enkla egenskaper redigeras i verktygsraden när en grupp av samma sorts komponent är markerad. Om markeringen innehåller en blandning av olika sorters komponenter (såsom motstånd och kondensatorer), visas inga egenskaper för redigering.

Vissa komponenter har mer avancerade egenskaper som inte är tillgängliga via verktygsraden. De finns i sidoraden **Objekt** till höger. Dioden har exempelvis ett antal olika beteendekaraktäristika som kan redigeras där.

Det finns en sorts egenskap som inte kan redigeras antingen via verktygsraden eller objektsidoraden, flerraderstext. Att dubbelklicka på objektet visar en dialogruta där texten kan matas in.

3.4 Simulering

Normalt kör simulering när en ny krets skapas. Simuleringens status visas nere till höger i kretsvyn, och kan ändras via menyn **Verktyg**. Först en kort förklaring av hur simulatorn fungerar. Det bör hjälpa dig att få ut så mycket som möjligt av den.

När en krets skapas eller ändras, delas de påverkade områdena upp i grupper av anslutningar och kopplingar som kan anses vara oberoende. Varje grupp simuleras sedan som en separat enhet (även om de fortfarande påverkar varandra via komponenterna), med tillhandahållen simulering beroende på gruppens komplexitet. Komplexa grupper, såsom de som innehåller ickelinjära komponenter som LED:ar, är långsamma att simulera. Grupper som bara innehåller logiska anslutningar, där bara en bestämmer värdet av anslutningarna, är snabbast att simulera.

Simuleringens resultat tillhandahålls med flera olika grafiska metoder.

Komponenternas anslutningar visar spänningsstaplar. De är färgade orange för positiva spänningar och blåa för negativa spänningar. Deras längd beror på spänningsnivån, och deras bredd på strömstyrkan som flyter genom anslutningen. De kan stängas av på sidan **Allmänt** i dialogrutan **Inställningar**.

Att hålla musen över en anslutning eller koppling visar ett litet verktygstips som anger spänningen och strömmen på det stället i kretsen. Flera komponenter ger också grafisk återmatning, till exempel LED:ar, voltmätare och amperemätare.

Till sist finns oscilloskopet, som beskrivs i nästa avsnitt.

3.5 Oscilloskop

Oscilloskopet kan spela in logisk, spännings- och strömdata. Logikproben är optimerad för att lagra Booleska samplingar, och bör användas istället för spänningsproben vid mätning av logik.

Skapa en ny probkomponent och anslut den på lämpligt ställe i kretsen för att samla in data. Utdata ritas omedelbart upp i oscilloskopet. Genom att lägga till flera prober, trycks utdata ihop

intill varandra. Det går att flytta om dem genom att dra pilarna till vänster om oscilloskopskärmen och ändra deras färg via probens egenskaper.

För spännings- och strömprober, kan ingångsvärdenas intervall justeras i sidoraden **Objekteditor** till höger.

Zoomning styrs av ett skjutreglage. Skalningen är logaritmisk. För varje bildpunkt som reglaget flyttas, multipliceras zoomfaktorn med en konstant. KTechlab simulerar logik till en maximal precision av 1 mikrosekund, och i maximal zoomnivå motsvarar en mikrosekund 8 bildpunkter.

När rullningslisten dras till slutet förblir den där när ny data spelas in. Annars förblir rullningslisten vid en fast tid. Oscilloskopskärmen kan också flyttas framåt och bakåt genom att vänsterklicka och dra visningen. På grund av begränsningar i det underliggande grafiska komponentsystemet är panoreringen mycket grynig vid största zoomning.

Att högerklicka på oscilloskopskärmen visar en meny där det går att bestämma antal gånger som oscilloskopskärmen uppdateras. Det tillåter antingen en jämnare visning, eller reducerad processoranvändning.

3.6 Delkretsar

Delkretsar erbjuder ett återanvändbart och snyggt sätt att använda en krets, när man bara är intresserad av att interagera med kretsens externa anslutningar. Delkretsen skapas som en integrerad krets, IC, där anslutningarna tillhandahåller interaktion med den interna kretsen.

Först måste kretsen som ska användas som en mall för att skapa en delkrets konstrueras. Interaktionsställen definieras med komponenten **Extern anslutning**. De måste kopplas in och placeras där du vill att anslutningen ska vara placerad på delkretsens integrerade krets.

Markera därefter komponentgruppen och de externa anslutningarna som ska göras till en delkrets, och välj **Skapa delkrets** i högerklicksmenyn. Du blir tillfrågad om att ge delkretsen ett namn. När den väl har skapats, dyker namnet upp i väljaren **Komponenter** under avdelningen **Delkretsar**. De kan behandlas som vilken vanlig komponent som helst, med det ytterligare alternativet att ta bort dem genom att högerklicka på objektet och välja **Ta bort**.

Kapitel 4

FlowCode

4.1 Inledning

FlowCode tillåter mycket snabb och enkel konstruktion av ett PIC-program. Efter att användaren har skapat ett flödesschema från tillgängliga programdelar, kan KTechlab konvertera flödesschemat till ett antal olika format. För att exempelvis mata ut hexkod, sker följande kedja av konverteringar:

1. FlowCode konverteras till Microbe, ett högnivåspråk vars kompilator distribueras med KTechlab.
2. Det körbara programmet **microbe** kompilerar Microbe-filen till PIC-assembler.
3. Till sist, tar **gpasm** PIC-assemblerfilen och matar ut programmets hexkod.

Om du inte har installerat gputils, som **gpasm** distribueras med, kan naturligtvis inte det sista steget utföras.

4.2 Skapa ett program

Varje FlowCode-program behöver en unik startpunkt. Det är stället som programmet körs från när PIC:n startas. För att definiera punkten, öppna sidoraden med flödeselement till vänster, och dra över elementet **Start**. KTechlab låter dig bara använda en av dem.

Därefter kan programmet konstrueras genom att använda de fördefinierade elementen till vänster, eller infoga egen kod (med assembler- eller Microbe-format) via elementet **Inbädda**. Programflödet kontrolleras via anslutningarna mellan flödeselementen. Avsnitt 3.2 ger mer detaljerad information om hur anslutningar skapas.

FlowCode påtvingar begränsningar förutom de i en krets rörande vad som kan anslutas. Exempelvis kan varje flödeselement bara ha en utgående anslutning. Ytterligare begränsningar beskrivs i Avsnitt 4.4.

4.3 PIC-inställningar

När ett nytt FlowCode-dokument skapas, visas en bild av PIC:n som används i arbetsområdets övre vänstra hörn. Den representerar de ursprungliga inställningarna av PIC:n.

Varje anslutning som visas på bilden av PIC:n anger anslutningens ursprungliga typ (ingång eller utgång) och dess ursprungliga tillstånd (hög eller låg). De går att ändra genom att dra anslutningen för att ange dess typ och klicka på den för att ändra dess tillstånd.

Dialogrutan **Inställningar** som visas genom att klicka på knappen **Inställningar**, gör det också möjligt att redigera de ursprungliga anslutningstyperna och tillstånden, i detta fall genom att redigera de binära värden som skrivs till PORT- och TRIS-registren. Förutom anslutningsinställningarna, gör dialogrutan det också möjligt att redigera de ursprungliga variabelvärdena i PIC-programmet.

Längst ner finns en lista över anslutningsavbildningar som för närvarande är definierade, samt knappar för att hantera dem. Anslutningsavbildningar används för att ange hur **7-segment** eller **Knappsats** ansluts till en PIC. För att använda FlowCode-elementen 7-segment eller knappsats, måste en anslutningsavbildning först definieras här.

4.4 Nästla FlowCode

Många flödeselement, såsom subrutiner och snurror, kan innehålla egen kod. Efter ett sådant omgivande element har skapats kan flödeselement läggas till genom att antingen dra eller släppa dem i det omgivande elementet. Det omgivande elementet markeras för att ange att det används som omgivning för flödeselementet.

Det omgivande elementet är ansvariga för flödeselement nästlade inne i det. Om expansionsknappen avmarkeras, döljes alla ingående flödeselement, och på motsvarande sätt visas innehållet när expansionsknappen klickas igen. Kopplingar kan inte göras mellan flödeselement i olika omgivande element, och innehållet i ett omgivande element flyttas omkring tillsammans med det.

Kapitel 5

Microbe

5.1 Inledning och generell syntax

Microbe kompilerar program skrivna i språket anpassat för PIC:er, som ett medföljande program till KTechlab. Syntaxen har skapats för att passa ett FlowCode-program. Syntaxen för att köra **microbe** på kommandoraden är:

```
microbe [väljare] [indata.microbe] [utdata.asm]
```

där väljare är:

- `--show-source`: Lägger till varje Microbe-källkodsrad som en kommentar i assemblerutmatningen innan själva assemblerinstruktionerna för den raden.
- `--no-optimize`: Förhindra optimering av instruktionerna skapade från källkoden. Optimering är oftast säker, alltså är väljaren i huvudsak avsedd för avlusning.

Filen indata.microbe måste identifiera PIC:n som är målet genom att infoga namnet på PIC:n längst upp i filen, t.ex. är namnet på PIC16F84 ″P16F84″.

Example 5.1 Enkelt fullständig Microbe-program

```
P16F84

a = 0
repeat
{
        PORTA = a
        a = a + 1
}
until a == 5

end
```

5.1.1 Namngivningskonventioner

Följande regler gäller för variabelnamn och etiketter:

- De kan bara innehålla alfanumeriska tecken [a..z][A..Z][0..9] och understrecket ”_”.
- De är skiftlägeskänsliga.
- De kan inte börja med en siffra.
- De får inte börja med ‘__’ (dubbla understreck) eftersom det är reserverat för användning av kompilatorn.

5.1.2 Parenteskonventioner

Klammerparenteser, {}, anger början och slutet på ett kodblock. De kan finnas var som helst före början och efter slutet på kodblocket. Exempel på korrekta kodblock:

```
sats1 {
        någon kod
}
```

```
sats2 {
        annan kod }
```

```
sats3
{
        annan kod
}
```

```
sats5 {
        kodblock
} sats6
```

5.1.3 Kommentarer

Kommentarer liknar C. // kommenterar bort resten av raden. /* och */ anger en flerraders kommentar.

```
// Det här är en kommentar
x = 2
/* Liksom denna
flerraders kommentar */
```

5.1.4 Programstruktur

PIC-id måste infogas överst i programmet. Slutet på huvudprogrammet anges med ‘end’. Subrutiner måste placeras efter ‘end’.

5.1.5 Subrutiner

En subrutin kan anropas var som helst i koden. Syntax:

```
sub SubrutinNamn
{
        // Kod...
}
```

Subrutinen anropas med ‘call *SubrutinNamn*’.

5.2 Microbe språkreferens

5.2.1 if

Villkorliga hopp. Syntax:

```
if [uttryck] then [sats]
```

eller

```
if [uttryck] then
{
        [satsblock]
}
```

På liknande sätt för else:

```
else [sats]
```

eller

```
else
{
        [satsblock]
}
```

Example 5.2 if

```
if porta.0 is high then
{
        delay 200
}
else
{
        delay 300
}
```

5.2.2 alias

Skapar ett alias från en sträng till en annan. Syntax:

```
alias [från] [till]
```

5.2.3 repeat

Kör satsblocket upprepade gånger tills uttrycket utvärderas till true. Utvärderingen av uttrycket utförs efter satsblocket, så att satsblocket alltid körs minst en gång. Syntax:

```
repeat
{
        [satsblock]
}
until [uttryck]
```

5.2.4 while

Liksom repeat, kör det satsblocket upprepade gånger. Dock utvärderas uttrycket innan körning, inte efter. Om uttrycket alltså utvärderas till false den första gången, körs inte satsblocket. Syntax:

```
while [uttryck]
{
        [satsblock]
}
```

5.2.5 goto

Gör att körning av koden fortsätter med nästa sats efter den angivna etiketten. Syntax för goto:

```
goto [etikettnamn]
```

Etikettsyntax:

```
etikettnamn:
```

Det är ofta god programmeringsvana att undvika användning av goto. Användning av kontrollsatser och subrutiner leder till ett mycket mer läsbart program.

Example 5.3 goto

```
goto MinEtikett

...

[MinEtikett]:
// Koden fortsätter på det här stället
```

5.2.6 call

Anropar en subrutin. Syntax:

```
call [SubrutinNamn]
```

där *SubrutinNamn* är namnet på subrutinen som ska anropas.

5.2.7 delay

Gör att körning av koden stoppas under angiven tidsperiod. Intervallet anges i millisekunder. Syntax:

```
delay [intervall]
```

NOT
För närvarande antar Microbe att PIC:n använder frekvensen 4 MHz, dvs. varje instruktion tar 1 mikrosekund att köra. Om det inte är fallet, måste intervallet justeras proportionellt.

5.2.8 sevenseg

Används för att definiera anslutningsavbildningen för en 7-segmentsdisplay (med gemensam katod) ansluten till PIC:n. Syntax

```
sevenseg [namn] [a] [b] [c] [d] [e] [f] [g]
```

där [a] ... [g] är PIC:ns anslutningar dit respektive segment i 7-segmentsdisplayen är kopplade. Anslutningarna kan antingen skrivas som PORTX.N eller RXN.

För att visa ett tal på 7-segmentdisplayen, behandlas anslutningsavbildningen som en enbart skrivbar variabel.

Example 5.4 Definiera och skriva ut på en 7-segmentsdisplay

```
sevenseg seg1 RB0 RB1 RB2 RB3 RB4 RB5 RB6
seg1 = x + 2
```

5.2.9 keypad

Används för att definiera anslutningsavbildningen för en knappsats ansluten till PIC:n. Syntax:

```
keypad [namn] [rad 1] ... [rad 4] [kolumn 1] ... [kolumn n]
```

där [rad 1] ... [rad 4] och [kolumn 1] ... [kolumn n] är PIC:ns anslutningar dit respektive rad och kolumn på knappsatsen är kopplade (för närvarande kan inte antal rader ändras). Se Avsnitt 5.2.8 (ovan) för mer information om anslutningsavbildningar.

Knappsatsens kolumner ska använda ett 100k pull-down motstånd till jord. Radanslutningarna måste vara konfigurerade som utgångar och kolumnanslutningarna som ingångar. När knappsatsen väl har definierats, behandlas den som en enbart läsbar variabel.

Example 5.5 Definiera och läsa från en knappsats

```
keypad keypad1 RB0 RB1 RB2 RB3 RB4 RB5 RB6
x = keypad1
```

Normalt är värdena som returneras för en knappsats:

- Värdet på siffran om det är en sifferknapp (1 till 3 längs övre raden, hexadecimalt A till D längs den fjärde kolumnen med fortsättning för varje extra kolumn).
- 253 för knappen i rad 4, kolumn 1.
- 254 för knappen i rad 4, kolumn 3.

Värdena kan definieras om genom att använda kommandot alias, där namnet på knappen i rad x, kolumn y (rader och kolumner börjar på 1), är Keypad_x_y. För att exempelvis ge asterisken på en 4x3 knappsats värdet noll, skulle följande alias användas:

Example 5.6 Skapa ett alias för en knapp på en knappsats till ett värde

```
alias Keypad_4_1 0
```

5.3 PIC in- och utmatning

5.3.1 Portriktning

Portriktningen anges genom att tilldela ett värde till TRIS*, där * är portbokstaven. Exempelvis:

Example 5.7 Ställa in portriktningar

```
TRISB = b'01111001'
```

Ovanstående ställer in anslutningarna RB1, RB2 och RB7 på PORTB som utgångar, och övriga anslutningar på PORTB som ingångar. I exemplet är b'01111001' den binära representationen av utgångstypen. Värdet 1 till höger representerar en utgång på RB0, och värdet 0 till vänster representerar en ingång på RB7.

5.3.2 Port in- och utmatning

Porten kan behandlas som en variabel, exempelvis:

Example 5.8 Skriva till en port

```
x = PORTA
```

Det ovanstående tilldelar värdet av PORTA till variabeln x.

5.3.3 In- och utmatning för anslutningar

Varje anslutning i en port erhålls genom att inleda anslutningsnumret med portnamnet: t.ex. anslutning 2 (med början på anslutning 0) på PORTA kallas *PORTA.0*. Syntaxen för att ställa in ett tillstånd på en anslutning är:

```
PORTX.N = STATE
```

där *STATE* kan vara *high* (hög) eller *low* (låg). Syntaxen för att testa en anslutnings tillstånd är:

```
if PORTX.N is STATEthen
```

Kombineras exemplen, får vi:

Example 5.9 Ställa in och prova anslutningens tillstånd

```
TRISA = 0
TRISB = 255
if PORTA.3 is high then
{
        PORTB.5 = low
}
else
{
        PORTB = PORTA + 15
}
```

5.4 Variabler

Alla variabler är 8-bitars heltal utan tecken, vilket ger intervallet 0 till 255. Microbe stöder de typiska unära (som arbetar med en variabel) och binära (som arbetar med två variabler) operatorerna som stöds av PIC:n. Dessutom stöder Microbe också division och multiplikation.

5.4.1 Unära operationer

- *rotateleft x* - Roterar variabeln x åt vänster via carry.
- *rotateright x* - Roterar variabeln x åt höger via carry.
- *increment x* - Ökar variabeln x med ett. Om x har värdet 255, slår den runt till 0.
- *decrement x* - Minskar variabeln x med ett. Om x har värdet 0, slår den runt till 255.

5.4.2 Aritmetik

Operationer som stöds:

- *Addition:* x + y
- *Subtraktion:* x - y
- *Multiplikation:* x * y
- *Division:* x / y
- *Binär exklusiv ELLER:* x XOR y
- *Binär OCH:* x AND y
- *Binär ELLER:* x OR y

5.4.3 Jämförelse

Operationer som stöds:

- *Lika med:* x == y
- *Skilt från:* x != y
- *Större än:* x > y
- *Mindre än:* x < y
- *Större än eller lika med:* x >= y
- *Mindre än eller lika med:* x <= y

Till exempel:

Example 5.10 Jämförelse

```
if PORTA
>= 5 then
{
        ...
}
```

Kapitel 6

Avlusning

6.1 Starta avlusaren

Avlusningsstöd tillhandahålls för assembler, SDCC och Microbe, när de öppnas som textdokument. Härifrån styrs stegning via menyn **Avlusa**. Det finns två sätt att starta avlusaren.

Om PIC-programmet redan kör i en krets, öppnas programmet genom att dubbelklicka på PIC-komponenten. För PIC-program i assembler, länkas avlusaren för det textdokumentet till PIC-komponenten. I detta fall kan inte avlusningsmenyn stoppa PIC-programmet, eftersom det ägs av PIC-komponenten.

Om assemblerfilen redan är öppen kan avlusaren köras via menyn **Avlusa**. Efter att ha kompilerat programmet är avlusaren redo med PIC-programmet pausat på den första instruktionen. Observera att när högnivåspråk avlusas, visas inte det nuvarande körningsstället om det inte finns någon rad som motsvarar den första assemblerinstruktionen som ska köras. I detta fall, kommer körningsstället till den första raden i programmet genom att klicka på **Nästa**.

6.2 Styra avlusaren

Avlusaren kan vara i ett av två lägen: körning och stegning. Medan det kör, simuleras PIC-programmet i realtid. För att tillåta stegning måste PIC-programmet pausas, antingen genom att klicka på **Avbryt** i menyn **Avlusa** eller klicka på pausknappen på PIC-komponenten.

I stegningsläge indikerar en grön pil i textdokumentets marginal nästa rad som ska köras (välbekant för användare av KDevelop). Det kan vara användbart att sätta på ikonkanten via menyn **Visa** (den kan sättas på permanent via dialogrutan **Editorinställningar**).

Det finns tre sorters stegning:

- **Stega**: Kör nuvarande instruktion. Den gröna pilen flyttas till nästa rad som ska köras.
- **Stega över**: Om nästa instruktion som ska köras är ett anrop eller liknande, stegar det här ″över″ anropet, och returnerar till stegningsläge när anropet väl har returnerat. Annars beter sig stega över identiskt med stega. Tekniskt sett lagras den ursprungliga stacknivån, och programkörningen pausas när stacknivån väl har återgått till sin ursprungliga nivå.
- **Steg ut ur**: Om den nuvarande körningen är inne i ett anrop eller liknande, väntar det här tills anropet returnerar. I likhet med att stega över, motsvarar det att vänta tills stacknivån återgår till ett mindre än den ursprungliga nivån, om den ursprungliga nivån är större än noll.

Brytpunkter gör det möjligt att pausa körningen när PIC-programmet når en angiven instruktion. För att ändra en brytpunkt på raden som innehåller markören, använd antingen menyn **Avlusa**, eller klicka på textdokumentets ikonkant.

Sidoraden **Symbolvisning** till höger, visar värden på speciella funktionsregister. För att ta reda på värdet på en variabel i de generella registren kan musen hållas över variabelnamnet i en instruktion som använder det registret. Observera att valet av bas i **Symbolvisning** bestämmer också hur värdet visas när musen hålls över en variabel.

Kapitel 7

Vanliga frågor

1. *KTechlab använder mycket processorkraft*

   Det finns flera möjliga orsaker. Simulering av kretsar som både innehåller reaktiva och icke-linjära komponenter (som kondensatorer och transistorer) kräver mycket processortid. Det går att pausa och återuppta simuleringen via menyn **Verktyg**.

   Att rita upp arbetsområdet (i synnerhet rita om många snabbt uppdaterade spänningsstaplar på anslutningar) kräver också mycket processorkraft. Det går att reducera uppdateringsfrekvensen, eller stänga av spänningsstaplarna via dialogrutan **Inställningar**. Oscilloskopets uppdateringsfrekvens kan också reduceras genom att högerklicka på dess skärm.

   Observera att nästa huvudutgåva av KTechlab kommer att vara mycket snabbare på att visa arbetsområdet och simulera reaktiva och icke-linjära komponenter.