DataBoard

7026

MPBASIC I

Beskrivning

MPBASIC I från Programteknik AB är en avancerad realtids heltals-Basic för användning på DataBoard Z80 SI0 enkortsdator 1004. MPBASIC I är ett snabbt och billigt sätt att utveckla mät- och styrapplikationer och kan även användas för undervisning i realtidsprogrammering.

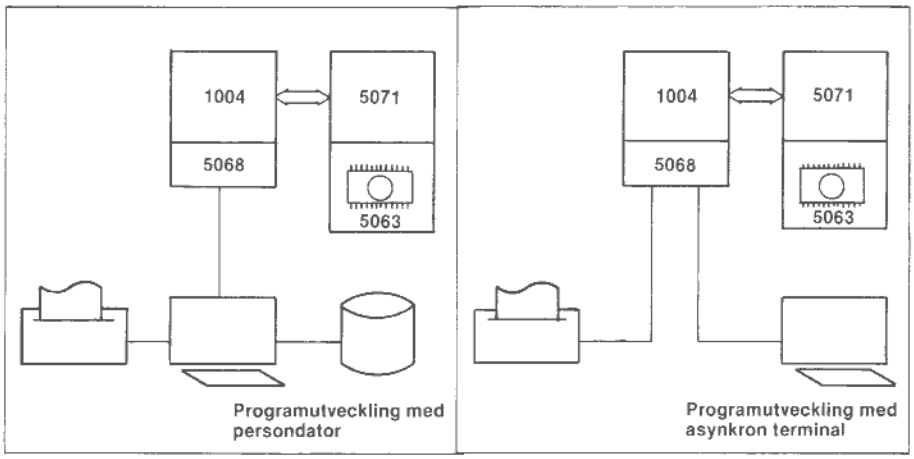
Ett vanligt problem i utveckling av industriella mikrodatortillämpningar är de höga initialkostnaderna, både i form av utrustning och kompetens. MPBASIC I kräver inget utvecklingssystem utan körs direkt i målsystemet där det också utprovas. Dessutom är MPBASIC I, med sina koncentrerade kommandon och funktioner, lätt att lära och en begränsad kunskap om programmering och datorteknik är i de flesta fall tillräcklig för att man skall nå bra resultat.

Programutvecklingen görs direkt på enkortsdatorn som i utvecklingsskedet bestyckas med ett RAM-minne samt en speciell MPBASIC I utvecklingstolk i EPROM. Den genererade koden är PROM-bar. Vid exekvering av programkod lagrad i PROM ersätts utvecklingstolken med en run-only tolk.

MPBASIC I har snabb kodexekvering, kräver relativt litet minnesutrymme och innehåller ett flertal specialfunktioner för hantering av data på bitnivå.

I många fall kan MPBASIC I ses som ett fullgott alternativ till assembler.

Sammanfattningsvis har MPBASIC I egenskaper som gör den till ett lättarbetat och billigt utvecklingshjälpmedel som snabbt och med små ekonomiska insatser leder fram till färdiga applikationer.



Programutveckling

Programutvecklingen görs direkt på enkortsdatorn 1004 som bestyckas med en 8 kbyte RAM samt en utvecklingstolk i 8 kbyte EPROM. Dessa monteras i två av de tre tillgängliga ByteWyde-hållarna på enkortsdatorn.

8 k RAM	Program Data
8 k EPROM 2764	Utvecklingstolk

När programmet är uttestat och har önskad funktion överförs programmet till EPROM. MPBASIC I är av semikompilerande typ vilket innebär att det är en kompaktkod för interpretering som lagras i minnet. Utvecklingstolken byts ut mot en run-only tolk och den använda 8 kbyte RAM-kretsen kan om så önskas ersättas med en 2 kbyte RAM-krets. Detta är framför allt av intresse när man använder zero-power-RAM med inbyggda batterier som ger spänningssäkert dataminne.

2 k RAM	Data
8 k EPROM 2764	Program
8 k EPROM 2764	Run-only tolk

Den utrustning som behövs förutom tidigare nämnda enkortsdator och MPBASIC I, som levereras med såväl utvecklingstolk som run-only tolk, är antingen en persondator eller en asynkron terminal, en 5068 adapter (till 1004) samt en PROM-programmerare.

Vid användning av asynkron terminal överförs programkoden till EPROM med PROM-programmeraren vilken ansluts till enkortsdatorns I/O-buss. Programmen kan förutom i körbar form även sparas i ASCII-format.

När en persondator används för programutveckling använder denna ett terminalprogram med filhantering vilken förutom lagring i EPROM även medger överföring till diskett eller ett annat massminne anslutet till persondatorn.

Programfunktioner

MPBASIC I medger att ett flertal processer (delprogram) kan exekveras till synes samtidigt. Detta sker genom tidsdelning och den styrande faktorn är ett specificerat antal programsatser som skall exekveras innan MPBASIC I skiftar över till nästa process. Varje process har

DATA INDUSTRIER AB

Box 2029, 18302 TÄBY. Tel 08-768 06 60. Telex 10978.

en egen variabeluppsättning, A-Z, men har samtidigt tillgång till ett antal gemensamma semaforer, A-Z. Semaforerna används för synkronisering mellan processerna och varje semafor kan förutom ett logiskt tillstånd, öppen/stängd, även ha ett numeriskt värde.

För gemensam datalagring finns en indexerad vektor som är tillgänglig från alla processer. För datalagring finns fyra FIFO'n eller PIPE's vilka också kan nås från samtliga processer. PIPE 0 och 1 har avbrottsstyrd inmatning från seriekanalerna på enkortsdatorn. Detta medger teckenmottagning med avbrottsstyrd buffring för senare bearbetning i en process. Varje PIPE har en längd om 80 byte.

Exekveringstiden för satser varierar mellan 0,14 ms och 2,3 ms.

Sammanfattning

Data

Talområde	Heltal -32768 till 32767. Långa heltal 0 till 4294967295
Variabler	26 st per process: A till Z
Semaforer	26 st: A till Z
Vektorer	1 st: E' (x). Storlek implementationsberoende

Funktioner

ABS (x):	Beloppet av uttrycket x
BIT (x,y):	1 om bit x i uttryck y är satt, annars 0
HIGH (x)	= = LOW (SWAP(X))
INP (x)	Värde läst från port x
LOW (x)	= = x AND 255 (de 8 låga bitarna i x)
PEEK (x)	Värdet i minnesadress x
PEEK2 (x)	Värdet (16 bitars) i minnesadress x,x + 1
RES (x<,x..>,y)	Värdet av uttryck y med bit(ar) x nollade
SEM (s)	Värdet av semafor s
SET (x<,x..>,y)	Värdet av uttryck y med bit(ar) x ettställda
SIZE	Ger storlek på återstående RAM för processen
SWAP (x)	Värdet av x med höga och låga åtta bitar bytta
TEST (s)	Ger 1 om semafor s är öppen, annars 0
WAIT (s)	Ger värdet av semafor s. Om semaforen var öppen vid anrop så stängdes den, annars fick processen vänta på att någon signalerade på semaforen
'T'	Ger ASCII-värdet av T

Operatorer

OR XOR AND	Bitvis
= > < > = < = < > + — * /	1 om sant, 0 om falskt
MOD	Rest vid division

Print-funktioner

CHR$ (x<,x..>)	Skriver tecken med ASCII-värden x
HEX$ (x<,x..>)	Skriver ut bytes i ASCII-hex (utan mellanrum)
LONG(l)	Skriver ut 32-bitsvariabel
"Text"	Skriver ut Text

Satser

En rad kan vara numrerad från 1 till 65535. Det kan finnas flera satser på en rad åtskilda med kolon ":".

CLRSEM s	Stäng semafor
DEC v	Minska variabel med ett
DJNZ v,r	Minska variabel med ett, om den är skild från 0 gå till rad r
FETCH <§v,>v	Om det finns en karaktär tillgänglig, sätt v till ASCII-värdet, annars sätt v till -1
GETCH <§c,>v	Vänta tills en karaktär finns tillgänglig. Tilldela variabel v ASCII-värdet av karaktären
GOSUB r	Gå till rad r men kom ihåg varifrån du gick
GOTO r	Gå till rad r
IF e THEN	Om uttrycket e är skilt från 0 utför det som kommer efter THEN
INC v	Öka variabel med ett
LADD I1,I2	Addera två långa variabler och lägg resultatet i I1
LDEC I	Minska en lång variabel med 1
LDIV I1,I2	Dividera två långa variabler, lägg kvoten i I1 och resten i I2
LET v = e	Lägg värdet av uttrycket e i variabel v
LINC I	Öka en lång variabel med 1
LMUL I1,I2	Multiplicera två långa variabler och lägg resultatet i I1
LSUB I1,I2	Subtrahera I2 från I1 och lägg resultatet i I1
OUT p,d<,p,d..>	Skicka data d till port p
POKE a,d<,d..>	Lägg ner data d i minnet på adress a
PRINT <§c,>	Skriv ut tal, text etc till kanal c
PRIORITY e	Sätt prioritet på process, dvs hur länge den får exekvera innan nästa process tar över
REM text	Kommentar
RETURN	Återvänd till senast exekverade GOSUB
SEM s = e	Lägg värdet av uttryck e i semafor s
SIGNAL S<,e>	Öppna semafor s (och lägg värdet av utryck e i den)
SLEEP e	Vänta i det antal femtedels sekunder som anges av uttrycket e
STOP	Avbryt exekvering av process
WAIT s	Vänta tills semafor s öppnas, stäng den omedelbart och exekvera

DIAB 504