bcc	title	prefix/class-number.revision
	SPECIAL AND BRANCH CONDITIONS IN CPU1	BCCPU1/S-10.2

checked	authors	approval date	revision date
Tom Hodge	Charles Simonyi	9/10/69	
checked		classification: Specification	
approved: Mel		distribution: Company Private	pages: 14

ABSTRACT and CONTENTS

This document contains the precise definition of special and branch conditions used by the CPU1 μcode.

1. The special and branch conditions in the CPU use the following classes of quantities:

1.1 Registers and busses in the μprocessor: X, Y, LB, M, Z, RØ, and E1 (LB is the output of the left Boolean box).

1.2 External signals: RS1 (SP), TO, STEP, PROCEED, RS2 (FAILURE SIGNAL) and STOP (as described in section 2).

1.3 A set of latches, each capable of holding a single bit of information: A, B, C, D, E, XOV, XTOV, XCAR, CARRY, OVFLW, M940, LOCK.

1.4 Two bits CY and OV, functions of the μprocessor adder.

1.5 Registers and busses of the physical map: SR, AD.

2. External signals to the CPU.

2.1 RS1 (Switch Processes).

The RS1 latch in CPUi is set if a μprocessor executes a REQUEST STROBE special condition with bit 4+i set in the X bus. RS1 can be reset by RRSL1 (23).

2.2 TO (timer overflow)

TO is the most significant bit in the interval timer.

2.3 STEP (single step mode)

The test processor can set this latch. The CPU may test it which clears it. STEP and PROCEED will be used for debugging purposes.

2.4 PROCEED (step or run depending on STEP)

The test processor can set PROCEED, the CPU can test it, which clears it.

2.5 RS2 (failure signal)

In case of a power failure or similar trouble (not as yet

specified) RS2 is set to 1.

2.6 STOP (stop or restart system)

This signal generated by the STOP(75) special condition sets the O registers of each μprocessor in the system to Ø.

3. The following special- and branch conditions described in the μprocessor specifications are used in the CPU:

MSØ-MS5	Name Used in μcode
Ø	no activity
1-12	shifts
13	SKZ (scratch pad with address in Z)
14	ALERT
15	POT
16	PIN
17	REQUEST STROBE #1
2Ø	UNPROTECT
21	unusable
22	LOAD MEMORY REQUEST PRIORITY FIELD (LPF)
23	RESET REQUEST STROBE LATCH #1 (SP)
24	RESET LOCAL & CENTRAL MEMORY REQUEST
25	REQUEST PROTECT
MCØ-MC5	
Ø	never branch

1,2Ø ALWAYS BRANCH

4. Special- and branch conditions not mentioned in the μprocessor specs are defined as follows:

* <name>

MS = <MSØ-MS5>: <SPL equivalent>
RETURN;

or

* <name>

MC = <MCØ-MC5>: RETURN(<SPL expression>);

with obvious meaning.

5. The SPL code:

```
*  μPROCESSOR REGISTERS AND BUSSES
   DECLARE X, Y, LB, M, Z, RØ, E1;
*  BØ-B23, 24 BITS IN A WORD
   MACRO BIT(XX) ← DECLARE FIELD B(Ø:XX,XX);
   BIT(Ø); BIT(1); BIT(2); BIT(3); BIT(4); BIT(5); BIT(6);
   BIT(7); BIT(8); BIT(9); BIT(1Ø); BIT(11); BIT(12); BIT(13);
   BIT(14); BIT(15); BIT(16); BIT(17); BIT(18); BIT(19); BIT(2Ø);
   BIT(21); BIT(22); BIT(23);
*  LATCHES IN THE CPU
   DECLARE A,B,C,D,E,XOV,XTOV,XCAR,CARRY,OVFLW,M940,LOCK;
*  FUNCTIONS OF THE ADDER
   DECLARE CY,OV;
*
   DECLARE FIELD OPCODE(Ø:3,8);
*  EXTERNAL SIGNALS
```

```
  DECLARE SP, TO, STEP, PROCEED, ABORT;
*  THE PHYSICAL MAP:
*  DECLARE 128 REGISTERS 11 BITS EACH
  DECLARE ARRAY PMAP[128];
  DECLARE FIELD EF(Ø:13,13), DB(Ø:14,14), PMRO(Ø:15,15),
            MAPVAL(Ø:16,23);
*  BITS FOR RING CHECK
  DECLARE FIELD BITØ(Ø,6,6), BIT1(Ø:7,7), BIT2(Ø:8,8),
            BIT3(Ø:9,9), BIT4(Ø:1Ø,1Ø), BIT5(Ø:11,11);
  FUNCTION RING (ADDRESS);
*  1 = USER RING
*  2 = UTILITY RING
*  3 = MONITOR RING
  RETURN (1 IF NOT ADDRESS$BITØ = 1 ELSE
           2 IF NOT ADDRESS$BIT1 = 1 AND
          (ADDRESS$BIT4 = 1 AND ADDRESS$BIT5 = OR NOT
          (NOT ADDRESS$BIT2 = 1 AND NOT ADDRESS$BIT3 = 1))
          ELSE 3);
*  SOURCE AND ADDRESS REGISTER (PART OF PM)
  DECLARE SR, AD;
*  FIELDS IN THE VIRTUAL ADDRESS
  DECLARE FIELD VA1 (Ø:6,12), VA2(Ø:13,23);
*  FIELDS IN THE ABSOLUTE ADDRESS
  DECLARE FIELD ERR1 (Ø:Ø,Ø), RINGERR (Ø:1,1) ERR2 (Ø:2,2),
            PMROERR (Ø:3,3), AA (Ø:5,12);
*  OTHER BITS AND PIECES
  DECLARE RCE, W;
*
*
```

```
*  SETB

MS = 26B:  B ← 1; RETURN;


*  GSB

MS = 27B:  E1$B23 ← E;
           E1$B22 ← XOV;
           E1$B21 ← XTOV;
           E1$B20 ← XCAR;
           E1$B14 ← M940; RETURN;

*  SETA

MS = 30B:  A ← 1; RETURN;


*  CLEARA

MS = 31B:  A ← B ← Ø; RETURN;


*  RRSL2 (Failure Latch)

MS = 32B:  RSL2 ← Ø; RETURN;


*  MFETCH

MS = 34B:  MAP; FETCH; RETURN;                    CFW  22 Feb 73


*  SABCDEØ

MS = 33B:  SR ← Z; A ← B ← C ← D ← E ← Ø; RETURN;    CFW  22 Feb 73


*  SETC

MS = 35B:  C ← 1; A ← B ← D ← Ø; RETURN;


*  SETD

MS = 36B:  D ← 1; RETURN;


*  MAP

MS = 37B:  RCE ← 1 if RING(RØ) > RING(SR) ELSE Ø;

MAP1:      W ← PMAP [AD ← RØ$VAl];
           E1$AA ← W$MAPVAL;
```

```
            E1$ERR1 ← RCE OR W$EF;
            E1$RINGERR ← RCE;
            E1$ERR2 ← W$PMRO OR NOT W$DB;
            E1$PMROERR ← W$PMRO;
            E1VA2 ← RØ$VA2; RETURN;

*   MEMORY REFERENCES

MS = 40-47B: 40:  RELEASE; RETURN;
             41:  PRESTORE; RETURN;
             42:  STORE; RETURN;
             43:  STORE & HOLD; RETURN;
             44:  FETCH; RETURN;
             45:  FETCH & HOLD; RETURN;
             47:  PREFETCH; RETURN;


*   CLEARA

MS = 50B:   A ← B ← Ø;

*   MAPS S                                          CFW 22 Feb 73
    ~~MAP=SR~~   (MAP + SR)

MS = 51B:   RCE ← 1 IF RING(RØ) > RING(SR) ELSE Ø;

            SR ← RØ;

            GOTO MAP1;


*   SSOURCE

MS = 52B:   SR ← Z; RETURN;


*   ZTOMAP

MS = 53B:   PMAP [RØ$VA1] ← Z; RETURN;


*   MAPAD

MS = 54B:   E1 ← VA1(AD); RETURN;


*   PSB

MS = 55B:   E ← Z$B23;
            XOV ← Z$B22;
            XTOV ← Z$B21;
            XCAR ← Z$B2Ø;
            M940 ← Z$B14; RETURN;
```

```
*  READS

MS = 56B:  E1 ← SR; RETURN;


*  HROV

MS = 57B:  OVFLW ← OV; CARRY ← CY; RETURN;

*  ~~SBB~~ SETBB (Set Bank B)                                   CFW 22 Feb 73

MS = 6ØB:  BSELECT ← 1; RETURN;

*  ~~SBA~~ SETBA (Set Bank A)                                   CFW 22 Feb 73

MS = 61B:  BSELECT ← Ø; RETURN;

*  ~~CLEARMAP~~ CMAPS Clear (All Maps)                          CFW 22 Feb 73

MS = 62B:  PMAP[W]$EF ← 1 FOR W ← Ø BY 1 TO 127:  RETURN;

*  <UNDEFINED>

MS = 63B:

*  ODDWORD FETCHES

MS = 64B,65B:  64:  FETCH; RETURN;
               65:  FETCH & HOLD; RETURN;
                                                               CFW 22 Feb 73
*  ROV

MS = ~~65B~~ 66B:  XTOV ← OV; XOV ← XOV OR OV; XCAR ← CY; RETURN;


*  CLEARTOV

MS = 67B:  XTOV ← Ø; RETURN;


*  ABCDØ:

MS = 7ØB:  A ← B ← C ← D ← Ø; RETURN;


*  X23LØK

MS = 71B:  LOCK ← X$B23; RETURN;

*  ~~CLEAR~~ CLM940 M940.  (Clear M940)                         CFW 22 Feb 73

MS = 72B:  M940 ← Ø; RETURN;
```

```
*  CLEARXOV                                        CFW 22 Feb 73

MS = 73B:  XOV ← Ø; RETURN;

*  CLEARMAP
   ~~COMAP~~ (Clear own map)

MS = 74B:  PMAP[W]$EF ← 1 FOR W ← Ø BY 1 TO 127; RETURN;

*  STOP

MS = 75B:  STOP ← 1; RETURN;

*  <UNDEFINED>

MS = 76B:

*  <UNDEFINED>

MS = 77B:

*

*


*  Z = Ø

MC = 2B:   RETURN(Z=Ø);

*  Z ≠ Ø

MC = 3B:   RETURN(Z≠Ø);

*  Z < Ø

MC = 4B:   RETURN(Z<Ø);

*  Z >= Ø

MC = 5B:   RETURN(Z>=Ø);

*  Z > Ø

MC = 6B: RETURN(Z>Ø);
```

```
*  NZ[15]
MC = 7B:    RETURN(Z$B15=Ø);

*  NZ[16]
MC = 1ØB:   RETURN(Z$B16=Ø);
```

CFW 22 Feb 73

```
*  RØ < Ø
MC = 11B:   RETURN(RØ$BØ=1);

*  RØ >= Ø
MC = 12B:   RETURN(RØ$BØ=Ø);

*  Z ≤ Ø
MC = 13B:   RETURN(Z≤Ø);

*  Z[18]
MC = 14B:   RETURN(Z$B18=1);

*  NZ[18]
MC = 15B:   RETURN(Z$B18=Ø);

*  X ≥ Ø
MC = 16B:   RETURN(X$BØ=Ø);

*  X < Ø
MC = 17B:   RETURN(X$BØ=1);

*  M[1]
MC = 21B:   RETURN(M$B1=1);

*  LB = Ø  (LB$BØ thru LB$B23=Ø)
MC = 22B:   RETURN(LB=Ø);
```

```
*  LB ≠ Ø  (LB$BØ thru LB$B23≠Ø)
MC = 23B:  RETURN(LB≠Ø);

*  RØ[2]
MC = 24B:  RETURN(RØ$B2=1);

*  NRØ[2]
MC = 25B:  RETURN(RØ$B2=Ø);

*  NATLAT (resets upon testing)
MC = 26B: RETURN(ATLAT=Ø);

*  NSP
MC = 27B:  RETURN(RSL1=Ø);

*  NPROTECT
MC = 30B:  RETURN(PROTECT≠X);

*  NFAIL
MC = 31B:  RETURN(RSL2=Ø);

*  NA
MC = 32B:  RETURN(A=Ø);

*  A
MC = 33B:  RETURN(A=1);

* B
MC = 34B:  RETURN(B=1);

*  C
MC = 35B:  RETURN(C =1);
```

```
*  ATLAT  (resets upon testing)
MC = 36B:  RETURN(ATLAT=1);

*  OV
MC = 37B:  RETURN(XOV=1);

*  NM[13]
MC = 40B:  RETURN(M$B13=Ø);

*  INTRPT
MC = 41B:  RETURN((TO OR RSL1) AND LOCK' OR STEP OR RSL2);

*  NLMPE (Not local  memory parity error) (Reset upon testing)
MC = 42B:  RETURN(LMPE=1);

*  M940
MC = 43B:  RETURN(M940=1);

*  NCMPE (Not central memory parity error) (Reset upon testing)
MC = 44B:  RETURN(CMPE=1);

*  BP
MC = 45B:  RETURN(BREAKPOINT=2);

*  XCAR
MC = 46B:  RETURN(XCAR=1);                    CFW 22 Feb 73

*  D
MC = 47B:  RETURN(D=1);

*  M[5]
MC = 5ØB:  RETURN(M$B5=1);
```

```
*  AORB
MC = 51B:  RETURN(AVB=1);

*  Z = 4B7
MC = 52B:  RETURN(Z=4B7);

*  Y < Ø
MC = 53B:  RETURN(Y$BØ=1);

*  STINST
MC = 54B:  RETURN(M$OPCODE > 12B AND M$OPCODE < 20B);

*  LAX
MC = 55B:  RETURN((Z$B3=1) ≠ (RØ$OPCODE=47B));

*  RØ[13:23]                                    (CFW 22 Feb 72)
MC = 56B:  RETURN(RØ$VA2=Ø);

*  OVFLW
MC = 57B:  RETURN(OVFLW=1);

*  NSTEP (Clears upon testing)
MC = 60B:  RETURN(STEP=Ø);

*  CARRY
MC = 61B:  RETURN(CARRY=1);

*  ST
MC = 62B:  RETURN((RØ$B2=1) OR (A=1) OR (B=1) OR (D=1));

*  MULT
MC = 63B:  RETURN(CARRY=RØ$BØ);
```

```
*  INTRPT1
MC = 64B:  RETURN((TOVRSL1) AND LOCK' OR SL2);

*  <UNDEFINED>
MC = 65B:

*  RØ[1]
MC = 66B:  RETURN(RØ$B1=1);

*  NPROCEED  (clears upon testing)
MC = 67B:  RETURN(PROCEED=Ø);

*  (SPANL SP AND NOT LOCK)                    CFW 22 Feb 73
MC = 70B:  RETURN(RSL1 AND LOCK'=1);

*  M[9]
MC = 71B:  RETURN(M$B9=1);

*  TO
MC = 72B:  RETURN(TO AND LOCK'=1);

*  M[Ø]=M[1]
MC = 73B:  RETURN(M$BØ=M$B1);

*  M[4]
MC = 74B:  RETURN(M$B4=1);

*  M[13]
MC = 75B:  RETURN(M$B13=1);
```

```
* FP

MC = 76B:   RETURN (E1←4  & 1   IF OVFLW = 1 ELSE
                    E1←5  & 1   IF Z = Ø AND LB = Ø ELSE
                    E1←6  & 1   IF Z = 4B7 AND LB = Ø ELSE
                          Ø     IF Z$BØ ≠ Z$B1 ELSE
                    E1←1  & 1   IF Z$B1 ≠ Z$B2 ELSE
                    E1←2  & 1   IF Z$B2 ≠ Z$B3 ELSE
                    E1←3  & 1);

* ⟨UNDEFINED⟩

MC = 77B:
```