EE 3755, Fall 2002

Solutions of HW#3

Solutions of HW#3, EE 3755

(1)

1 MA ← PC.

2 ADBUS = MA; MD ← DBUS; read = 1; PC ← INC(PC).

3 IR ← MD.

4 MA ← MD[4:15].

5 → ($\overline{IR[1]}$) / (26). "go to step 26 if ADD, OR, AND, MVT, JPL, JEQ, JMP, JSR".

6 → ($\overline{IR[0]} \vee \overline{IR[3]}$) / (16). "go to step 16 if CLC, SEC, ROR, RTS, NOT".

7 → ($\overline{IR[2]}$) / (10). "go to step 10 if MULT".

8 MD ← AC.

9 ADBUS = MA; DBUS = MD; write = 1; → (1). } "execution of MVF".

10 ADBUS = MA; MD ← DBUS; read = 1; R ← 16TO; d ← 0; CNT ← 4TO. "get multiplicand from memory and place it in MD. Initialize Booth algorithm".

11 → ($\overline{AC[15] \oplus d}$) / (15). "If 00 or 11 go to 15 to do shift"

12 → (AC[15]) (14). "If 10 go to 14 to do subtr."

13 ABUS = MD; BBUS = R; cin = 0; OBUS = ADD[1:16](ABUS; BBUS; cin); R ← OBUS; → (15). "add"

14 ABUS = $\overline{MD}$; BBUS = R; cin = 1; OBUS = ADD[1:16](ABUS; BBUS; cin); R ← OBUS. "subtr".

↳ continues on next page

```
15  R,AC,d ← R[0],R[0:15],AC[0:15];
    CNT ← INC(CNT); →(1/CNT, 1/CNT)/(1,11).
                                "shift and test counter".
16  →(IR[0])/(25). "If instr. is NOT go to 25".
17  →(IR[2])/(19). "If ROR or RTS go to 19".
18  cff ← IR[3]; →(1). "execute CLC, SEC".
19  →(IR[3])/(22). "go to step 22 if RTS"
20  CNT ← DEC(IR[12:15]).
21  AC[0] ← cff; AC[1:15] ← AC[0:14];
    cff ← AC[15]; CNT ← DEC(CNT);
    →(V/CNT, V/CNT)/(1,21).                  } "execution of ROR".

22  MA ← SP
23  ADBUS=MA; MD ← DBUS; read=1;
    SP ← INC(SP).                            } "execution of RTS".
24  PC ← MD[4:15]; → (1).

25  BBUS = AC; OBUS = BBUS; AC ← OBUS;       "execution of NOT"
    zff ← V/OBUS; nff ← OBUS[0]; →(1).
26  →(IR[0])/(29). "go to step 29 if JPL, JEQ, JNB, JSR".
27  ADBUS = MA; read=1; MD ← DBUS. "get operand from memory".
```

→ go to next page

```
28 ABUS = MD; BBUS = AC; cin = 0;
 OBUS = (ADD[1:16](ABUS; BBUS; cin) ! ABUS ∨ BBUS ! ABUS ∧ BBUS ! ABUS)
   * (¬IR[2] ∧ ¬IR[3], ¬IR[2] ∧ IR[3], IR[2] ∧ IR[3], IR[2] ∧ ¬IR[3]);
   AC ← OBUS; cff * (¬IR[2] ∧ ¬IR[3]) ← ADD[0](ABUS; BBUS; cin);
   zff ← ¬∨/OBUS; nff ← OBUS[0];
   vff * (¬IR[2] ∧ ¬IR[3]) ← (ABUS[0] ∧ BBUS[0] ∧ ¬ADD[1](ABUS; BBUS; cin))
                            ∨ (¬ABUS[0] ∧ ¬BBUS[0] ∧ ADD[1](ABUS; BBUS; cin));
   →(1). " Steps 27, 28 show the execution of
                ADD, OR, AND, MVT ".

29 →(IR[2])/(31). " Go to step 31 if JMP or JSR ".
30 PC * ((¬nff ∧ ¬IR[3]) ∨ (zff ∧ IR[3])) ← MA;
   →(1). " execution of JPL, JEQ ".
31 →(¬IR[3])/(33). " go to step 33 if JSR ".
32 PC ← MA; →(1). " execution of JMP ".
33 IR[4:15] ← MA.
34 MA ← DEC(SP); SP ← DEC(SP).
35 MD ← 4TO, PC.
36 ADBUS = MA; write = 1; DBUS = MD.
37 PC ← IR[4:15]; →(1).
```

" steps 33, 34, 35, 36, 37 represent the execution of JSR ".

Note: The above control sequence is not the most optimal one. You can definitely reduce the number of steps.

(4)

2 Consulting with the bits IR[5]; IR[6] and IR[10] of the shift/rotate instruction, (see page 2 of appropriate handout), one gets the following AHPL code for step 61:

```
61 AC[0] ← (AC[0]!AC[1]!AC[31]!cff!0)*
   (~IR[5]∧~IR[6]∧IR[10], ~IR[10], IR[5]∧IR[6]∧IR[10],
   IR[5]∧~IR[6]∧IR[10], ~IR[5]∧IR[6]∧IR[10]);
   AC[1:30] ← (AC[0:29]!AC[2:31])*(IR[10], ~IR[10]);
   AC[31] ← (AC[30]!AC[0]!cff!0)*
     (IR[10], IR[5]∧IR[6]∧~IR[10],
      IR[5]∧~IR[6]∧~IR[10], ~IR[5]∧~IR[10]);
   cff*(~IR[5]∧IR[6]) ← (AC[31]!AC[0])*
                        (IR[10], ~IR[10]);
   SHC ← DEC(SHC);
   → (∨/SHC, ~∨/SHC)/(61, 100).
```

[3]

0 "same as before; see appropriate handout"

1 BBUS=8T0, PC; OBUS=BBUS;
MA ← OBUS[8:31]; shf ← 0.

2 ADBUS=MA; read=1; MD ← DBUS;
ABUS=32T1; BBUS=8T0, PC; cin=0;
OBUS= ADD[1:32](ABUS; BBUS; cin);
PC ← OBUS[8:31].

3 ABUS=MD; OBUS=ABUS; IR ← OBUS

4, 5, 6 } "same as before; see appropriate handout".

7 ABUS = MD[8], MD[8], MD[8], MD[8], MD[8], MD[8], MD[8], MD[8], 24T0;
BBUS = $\overline{8T0}$, 24T0; OBUS = ABUS ∧ BBUS;
~~MDBE~~ MD[0:7] ← OBUS[0:7]; → (22).

8 " same as before

9 ABUS=MD; OBUS=ABUS;
MA ← OBUS[8:31].

10 " same as before"

11 ABUS=MD; BBUS=8T0,1X; cin=0;
OBUS=(ABUS ! ADD[1:32](ABUS; BBUS; cin)) *
($\overline{IR[6]}$, IR[6]);
MA ← OBUS[8:31].

12 " same as before"

13 ABUS=8T0,MA; OBUS=ABUS;
PC ← OBUS[8:31]; → (1).

14 "same as before".

15 ABUS=8T0, MA; OBUS=ABUS;
IR[8:31] ← OBUS[8:31]; → (32).

16 "same as before"

17 BBUS=AC; OBUS=BBUS; MD ← OBUS.

18 " same as before".

19 " " " " "

20 ABUS=MD; BBUS=(32T1 ! $\overline{32T0}$) *
($\overline{IR[3]}$, IR[3]);
cin=0;
OBUS=ADD[1:32](ABUS; BBUS; cin);
MD ← OBUS; zff ← $\overline{V|OBUS}$; nff ← OBUS[0].

21 " same as before".

22 " Same as the one shown on page 9 of the handout entitled : First example computer : Part (2)

30 } " Same as before; see appropriate handout"
31 }

32 BBUS = 8TO, SP ; ABUS = $\overline{32TO}$; cin = 0 ;
OBUS = ADD[1:32](ABUS ; BBUS ; cin) ;
MA ← OBUS[8:31] ; SP ← OBUS[8:31] .

33 BBUS = 8TO, PC ; OBUS = BBUS ; MD ← OBUS.

34 } "Same as before; see appropriate
35 } handout"

50 } " Same as before; see appropriate
51 } handout"

52 ABUS = (28TO, IR[12:15] ! $\overline{28TO}$, IR[12:15]) *
($\overline{IR[12]}$, IR[12]) ;
BBUS = (AC ! 8TO, IX ! 8TO, SP) *
($\overline{IR[12]}$, IR[12] ∧ $\overline{IR[13]}$, IR[12] ∧ IR[13]) ;
OBUS = (ABUS ! BBUS) * (IR[10], $\overline{IR[10]}$) ;
MD ← OBUS.

53 " Same as before; see appropriate handout"

The steps 60, 61, 62 shown below are the ones responsible for the shift/rotate instructions. Explicit BUS-connections are shown. The way shifting/rotating is accomplished is by connecting AC to the BBUS, shift/rotate the BBUS, connect the shifted/rotated BBUS to the OBUS and target AC from the OBUS. The step 62 is new and the purpose of its existence will be explained after steps 60, 61 and 62 are presented.

60 ABUS = 27T0, IR[11:15]; BBUS = $\overline{32T0}$;
cin = 0; OBUS = ADD[1:32](ABUS; BBUS; cin);
SHC ← OBUS[27:31].

→ steps 61, 62 on next page

61 BBUS = AC;
OBUS[0] = (BBUS[0] ! BBUS[1] ! BBUS[31] ! cff ! 0)
* (IR[5] ∧ IR[6] ∧ IR[10], IR[10],
IR[5] ∧ IR[6] ∧ IR[10], IR[5] ∧ IR[6] ∧ IR[10],
IR[5] ∧ IR[6] ∧ IR[10]);
OBUS[1:30] = (BBUS[0:29] ! BBUS[2:31]) *
(IR[10], IR[10]);
OBUS[31] = (BBUS[30] ! BBUS[0] ! cff ! 0) *
(IR[10], IR[5] ∧ IR[6] ∧ IR[10],
IR[5] ∧ IR[6] ∧ IR[10], IR[5] ∧ IR[10]);
AC ← OBUS;
cff * (IR[5] ∧ IR[6]) ← (BBUS[31] ! BBUS[0]) *
(IR[10], IR[10]).

62 ABUS = 27T0, SHC; BBUS = 32T0; cin = 0;
OBUS = ADD[1:32](ABUS; BBUS; cin);
SHC ← OBUS[27:31];
→ (V/SHC, V/SHC) / (61, 100).

Some explanations follow on the next page.

Explanations: The reason why I used a separate step 62 to be responsible for decrementing and testing SHC is because I assumed that the decrementation of SHC is performed by the adder of the ALU and the buses ABUS; BBUS and OBUS are involved. I assumed that the shift counter SHC is a connection to ABUS. If a dedicated decrementer was to be used for decrementing SHC in a way that the buses ABUS; BBUS; OBUS were not to be involved, then steps 61 and 62 could have been combined in one step.

The remaining steps 70 through 101 now follow

70
71
72
73
74
} "Same as before; see appropriate handout"

75 BBUS = 8TO, SP; ABUS = $\overline{32TO}$; cin = 0;
OBUS = ADD[1:32](ABUS; BBUS; cin);
MA ← OBUS[8:31]; SP ← OBUS[8:31].

76 BBUS = AC; OBUS = BBUS; MD ← OBUS.

77 "same as before; see appropriate handout".

78 BBUS = 8T0, SP; OBUS = BBUS;
MA ← OBUS [8:31].

79 ADBUS = MA; read = 1; MD ← DBUS;
BBUS = 8T0, SP; ABUS = 32T1; cin = 0;
OBUS = ADD [1:32] (ABUS; BBUS; cin);
SP ← OBUS [8:31].

80 ABUS = MD; OBUS = ABUS; AC * $\overline{\text{IR[8]}}$ ← OBUS;
PC * IR[8] ← OBUS [8:31]; → (100).

100 "same as before"

101 ABUS = IR; OBUS = ABUS;
IR[0:15] ← OBUS [16:31]; shf ← 1;
→ (50).