POCO の命令セット、ハードウェア構成は 2 ページ目以降に示しますが、問題中に不明な点がある場合や手元に資料がない場合は、各自判断して答え、その旨を記して下さい。

1. (1) 10 進数の-18 を 2 の補数表示の 2 進数で示せ。 18 は 00010010 11101101+1=11101110
(2) 40-18 を 2 の補数を用いることで計算せよ。途中結果も示せ。00101000+11101110=100010110 最上位の 1 は無視する。これは 22 に当たる。
(3) (2) の答えを 16 進数で示せ。　0x16

2. 16bit RISC POCO で下の命令を順に実行した。

```
LDHI r0, #0x01          01010(opcode) 000 00000001
ADDIU r0, #0x02         01101(opcode) 000 00000010
LD r2,(r0)              00000(opcode) 010 000 01001(funct)
ADDIU r0, #0x01         01101(opcode) 000 00000001
LD r3,(r0)              00000(opcode) 011 000 01001(funct)
ADD r2, r3              00000(opcode) 011 000 00110(funct)
ADDIU r0, #0x01         01101(opcode) 000 00000011
ST r2,(r1)              00000(opcode) 011 000 01000(funct)
```

a) 上記の機械語命令を示せ。またそれぞれの opcode フィールド、function(funct) フィールド (あれば) はどうなるかを示せ (右)
b) r0 の値がどのように変化するかを示せ。0x0100,0x0102,0x0103,0x0104
c) メモリ中のアドレスの何番地と何番地の内容が足されるか？また答えはどこの番地に格納されるか？ 0x0102 と 0x0103 答えは 0x0104 番地に格納される。

3 (1) 2 ページ目の POCO は、授業でやった構成をやや簡単にしたものである。a)OR 命令、b)LDIU 命令、c)BEZ 命令を実行する際に、各制御信号線をどのように設定すれば良いか？表に付け加えよ。

表 1: 各命令の制御信号

	comsel	alu_bsel	rf_csel	rwe	we	pcsel
OR	00	00	0	1	0	0
LDIU	01	10	0	1	0	0
BEZ	-	-	-	0	0	1(if Zero)

(2) JR 命令を取り付けるためには、上記表中のどの信号線に関連するハードウェアモジュールを改造する必要があるか？
pcsel 周辺

4-1. 0 番地から 100 番地までのメモリに全て 0 を書き込むプログラムを書け。

```
      LDI r0,#0
      LDI r1,#100
      LDI r2,#0
loop: ST r2,(r0)
      ADDI r0,#1
      ADDI r1,#-1
      BNZ r1,loop
end   BEZ r1,end
```

4-2. 4-1 のプログラムをサブルーチン化せよ。スタート番地を r0、0 を書き込む数の個数を r1 に入れて呼び出すようにせよ。

```
      LDI r2,#0
loop: ST r2,(r0)
      ADDI r0,#1
      ADDI r1,#-1
```

```
BNZ r1,loop
JR r7
```

4-3. 4-2 のプログラムで保存しなければならないレジスタはどれかを示せ。
r0,r1,r2

5-1.32Kword の主記憶に対して 512word のキャッシュを設けた。ブロックサイズは 8 ワードとした時、以下の構成のキャッシュの index と tag(key) の大きさを求めよ。　(a) ダイレクトマップキャッシュ
キャッシュ上には 512/8=64 ブロック載る。つまりインデックスは 6 ビットとなる。ブロック内アドレスは 3 ビットで、主記憶のアドレスは 15 ビット、つまりタグは 7 ビットとなる。
(b) 2-way セットアソシィアティブキャッシュ
深さ 32、幅 8 ビットのタグが 2 つ必要となる。
(c) 4-way セットアソシィアティブキャッシュ

深さ 16、幅 9 ビットのタグが 4 つ必要となる。
5-2. キャッシュミスの原因を三つ示し、それぞれ簡単に説明しなさい。
容量ミス、容量の不足によって起きる。競合ミス、インデックスが競合することにより起きる、初期化ミス、プログラム開始時やコンテキスト切り替え時にキャッシュ上に有効データが存在しないことにより起きる。

A) POCO の命令コード

```
NOP                                              00000 --- --- 00000
MV  rd,rs    rd <- rs                            00000 ddd sss 00001
AND rd,rs    rd <- rd AND rs                     00000 ddd sss 00010
OR  rd,rs    rd <- rd OR rs                      00000 ddd sss 00011
SL  rd       rd <- rd<<1                         00000 ddd --- 00100
SR  rd       rd <- rd>>1                         00000 ddd --- 00101
ADD rd,rs    rd <- rd + rs                       00000 ddd sss 00110
SUB rd,rs    rd <- rd - rs                       00000 ddd sss 00111
ST rs, (ra)     rs -> (ra)                       00000 sss aaa 01000
LD rd, (ra)     rd <- (ra)                       00000 ddd aaa 01001
LDI rd,#X       rd <- X (符号拡張)               01000 ddd XXXXXXXX
LDIU rd,#X      rd <- X (符号拡張なし)           01001 ddd XXXXXXXX
ADDI rd,#X      rd <- rd + X (符号拡張)          01100 ddd XXXXXXXX
ADDIU rd,#X     rd <- rd + X (符号拡張なし)      01101 ddd XXXXXXXX
LDHI rd,#X      rd <- X|0                        01010 ddd XXXXXXXX
BEZ rd, X   if (rd==0) pc <- pc + X              10000 ddd XXXXXXXX
BNZ rd, X   if (rd!=0) pc <- pc + X              10001 ddd XXXXXXXX
BPL rd, X   if (rd>=0) pc <- pc + X              10010 ddd XXXXXXXX
BMI rd, X   if (rd<0) pc <- pc + X               10011 ddd XXXXXXXX
JMP X       pc <- pc + X                         10100 XXXXXXXXXXX
JAL X       r7 <- pc, pc<- pc + X                10101 XXXXXXXXXXX
JR  rd              pc <- rd                     00000 ddd --- 01010
```

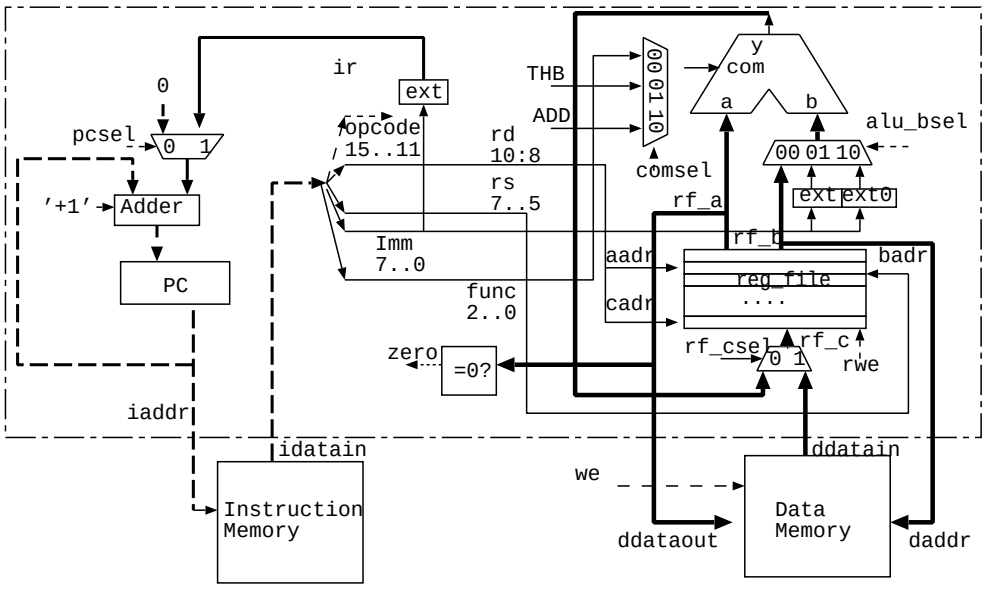

図 1: POCO のデータパス