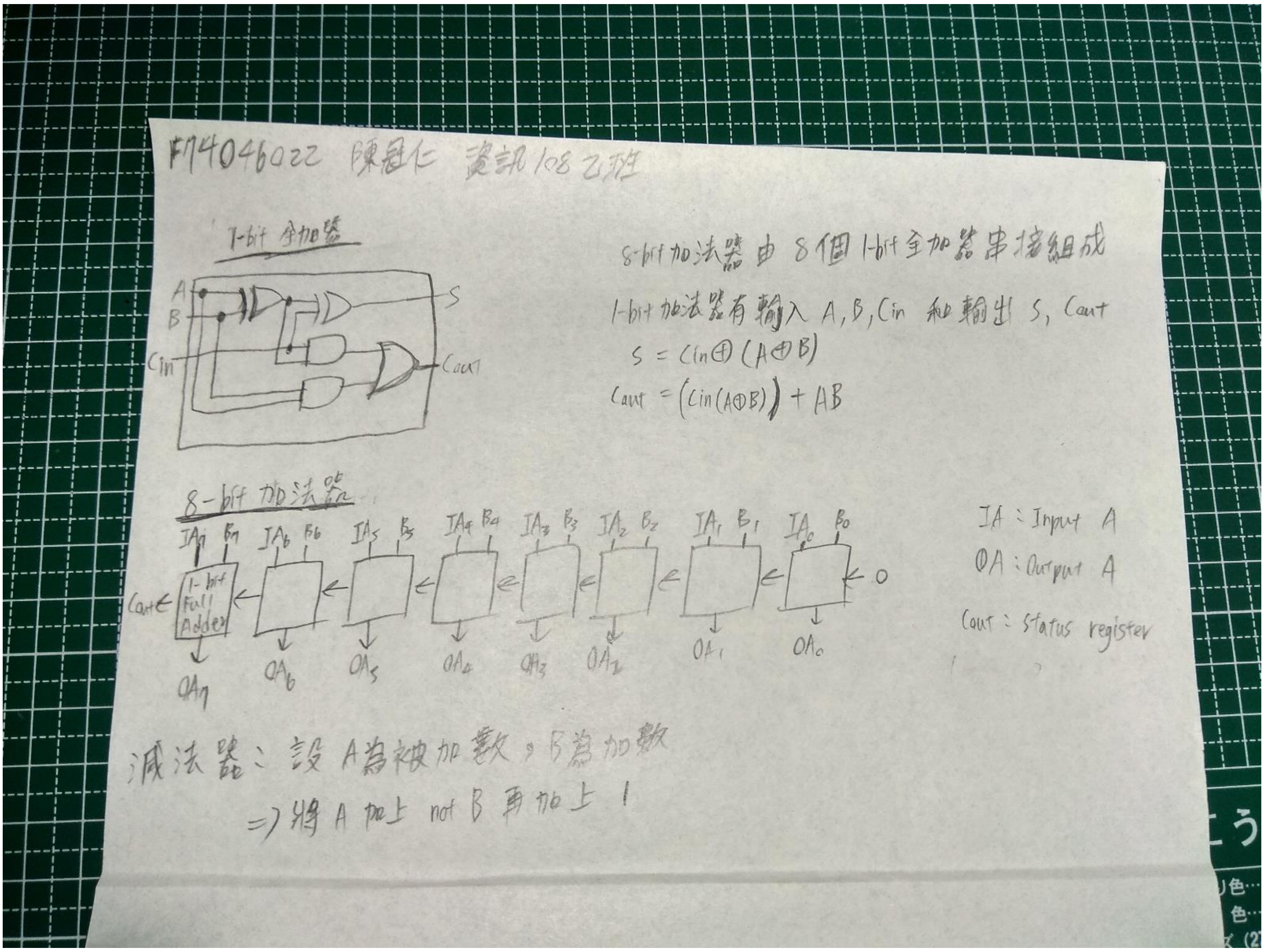
F74046022 陳冠仁 資訊 108 乙班

課堂能力測驗訂正上傳

小美的加法器



F74046022 陳冠仁 資訊108乙班

1-bit 全加器

8-bit 加法器由 8 個 1-bit 全加器串接組成

1-bit 加法器有輸入 A, B, Cin 和輸出 S, Cout

$S = Cin \oplus (A \oplus B)$

$Cout = (Cin(A \oplus B)) + AB$

8-bit 加法器

IA : Input A

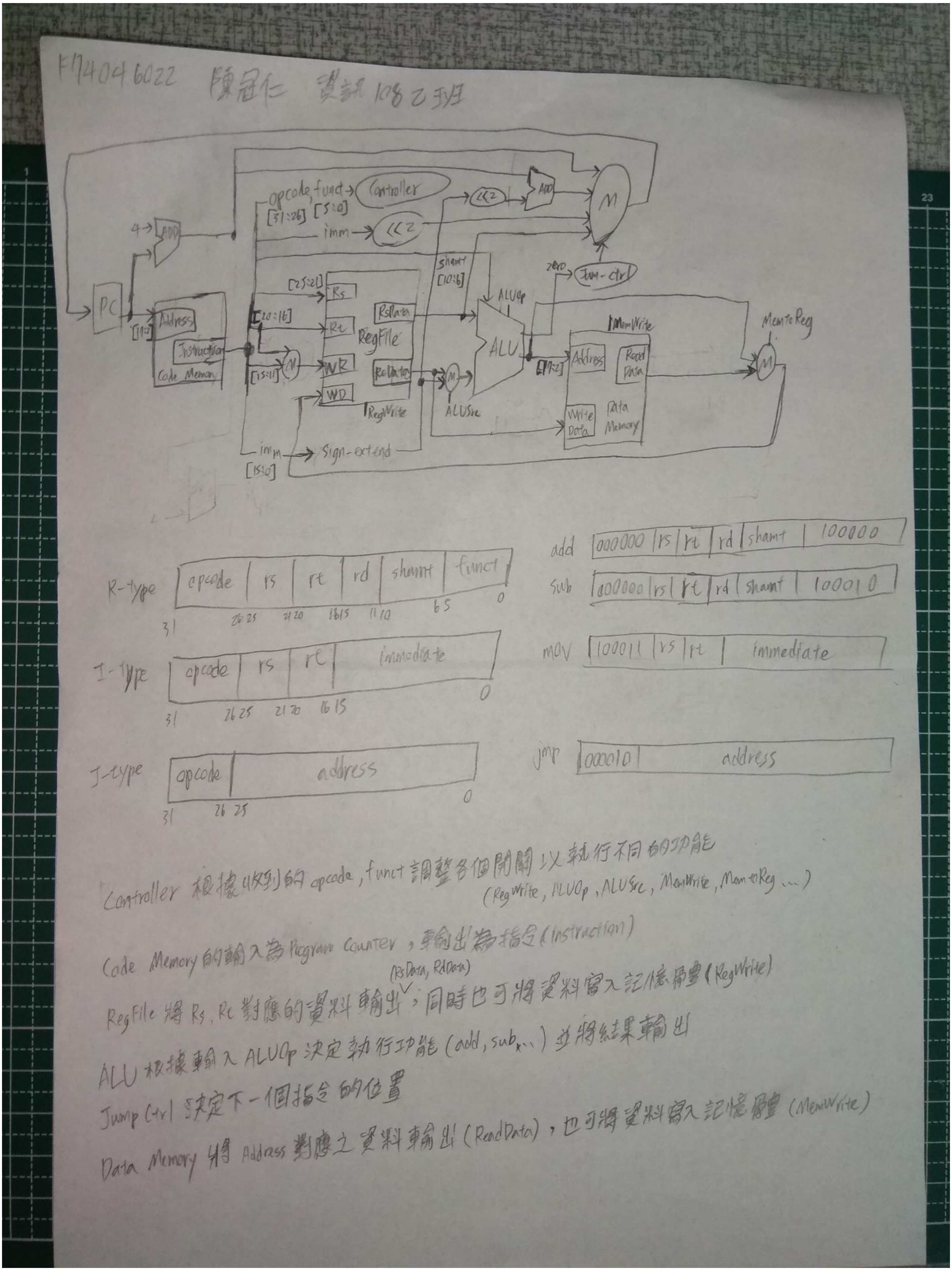
OA : Output A

Cout : Status register

減法器：設 A為被加數，B為加數

⇒ 將 A 加上 not B 再加上 1

小明的 CPU



F74046022 陳冠仁 資訊108乙班

R-type

opcode	rs	rt	rd	shamt	funct
31 26	25 21	20 16	15 11	10 6	5 0

I-type

opcode	rs	rt	immediate
31 26	25 21	20 16	15 0

J-type

opcode	address
31 26	25 0

add	000000	rs	rt	rd	shamt	100000
sub	000000	rs	rt	rd	shamt	100010
mov	100011	rs	rt	immediate		
jmp	000010	address				

Controller 根據收到的 opcode, funct 調整各個開關以執行不同的功能 (RegWrite, ALUOp, ALUSrc, MemWrite, MemtoReg ...)

Code Memory 的輸入為 Program Counter，輸出為指令 (Instruction)

RegFile 將 Rs, Rt 對應的資料輸出 (RsData, RtData)，同時也可將資料寫入記憶體 (RegWrite)

ALU 根據輸入 ALUOp 決定執行功能 (add, sub, ...) 並將結果輸出

Jump Ctrl 決定下一個指令的位置

Data Memory 將 Address 對應之資料輸出 (ReadData)，也可將資料寫入記憶體 (MemWrite)

小西的 C 程式

```
#include <stdio.h>
#include <stdlib.h>

typedef struct Node{
	char ch;
	struct Node* next;
}Node;

typedef struct Stack{
	char data[256];
	int pos;
}Stack;

char remove_from_llist(Node **llist, int n){
	Node *cur=*llist, *prev=NULL;
	char ch;

	if(n==0 && cur!=NULL){
		ch = (*llist)->ch;
		*llist = (*llist)->next;
		return ch;
	}

	while(cur->next!=NULL && n>0){
		prev = cur;
		cur = cur->next;
		--n;
	}

	if(n>0){
		printf("Index out of bound\n");
		return '\0';
	}else{
		prev->next = cur->next;
		ch = cur->ch;
		free(cur);
		return ch;
	}
}
```

```
void push_to_stack(Stack *stack, char ch){
	if (stack->pos>=255){
		printf("StackError: Stack is full\n");
		return;
	}
	stack->data[++stack->pos] = ch;
}

char pop_from_stack(Stack *stack){
	if (stack->pos<0){
		printf("StackError: Stack is empty\n");
		return '\0';
	}
	return stack->data[stack->pos--];
}

void print_stack(Stack *stack){
	int i;
	for(i=0; i<=stack->pos; ++i)
		printf("%c", stack->data[i]);
	printf("\n");
}

void print_llist(Node *llist){
	Node *cur;
	for(cur=llist; cur!=NULL; cur=cur->next)
		printf("%c", cur->ch);
	printf("\n");
}

int main(int argc, char* argv[]){
	char ch = (char)('a'-48);

	Stack *stack = (Stack*) malloc(sizeof(Stack));
	stack->pos=-1;

	Node *llist = (Node*) malloc(sizeof(Node));
	llist->ch = ch;

	Node *cur = llist;
	int i;
	for(i=0; i<50; ++i){
        cur->next = (Node*) malloc(sizeof(Node));
        cur = cur->next;
        cur->ch = ++ch;
    }

    printf("\nInitial State : (50 nodes in linked list, empty stack)\n");
    printf("Linked list:\t");   print_llist(llist);
    printf("Stack:\t\t");       print_stack(stack);

    printf("\nRemove 1st node from linked list and push it to stack\n");
    push_to_stack(stack, remove_from_llist(&llist, 0));
    printf("Linked list:\t");   print_llist(llist);
    printf("Stack:\t\t");       print_stack(stack);


    printf("\nRemove 10th node from linked list and push it to stack\n");
    push_to_stack(stack, remove_from_llist(&llist, 9));
    printf("Linked list:\t");   print_llist(llist);
    printf("Stack:\t\t");       print_stack(stack);

    printf("\nPop from stack\n");
    pop_from_stack(stack);
    printf("Linked list:\t");   print_llist(llist);
    printf("Stack:\t\t");       print_stack(stack);

    return 0;
}
```

stack 以 C 語言中的 struct 實作
其中包含由 256 個 char 組成的 array 和一個 integer 代表 stack 現在的位置
當 push 時朝高記憶體位置移動，pop 時朝低記憶體位置移動