國立台灣大學電機資訊學院電子工程學研究所

系統晶片設計實驗
Soc Design Laboratory

LabD Report

WB-SDRAM

學生： M11202109 蘇柏丞
M11202103 陳泓宇
M11202207 呂彥霖

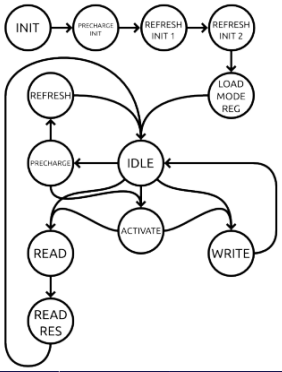
老師： 賴 瑾

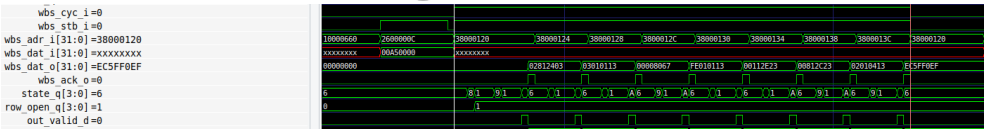
中華民國 112 年 12 月 17 日

一、 The SDRAM controller design and SDRAM bus protocol：

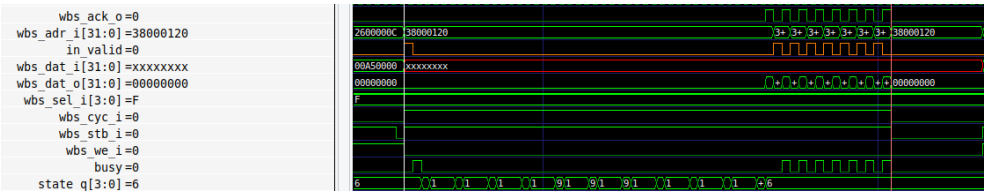
SDRAM Controller

- INIT→IDLE
- IDLE→ACTIVATE→WRITE→IDLE
- IDLE→ACTIVATE→READ→READ_RES →IDLE
- IDLE→WRITE→IDLE
- IDLE→READ→READ_RES→IDLE
- IDLE→PRECHARE→ACTIVATE→WRITE →IDLE
- IDLE→PRECHARE→ACTIVATE→READ →READ_RES→IDLE
- IDLE→PRECHARE→REFRESH→IDLE



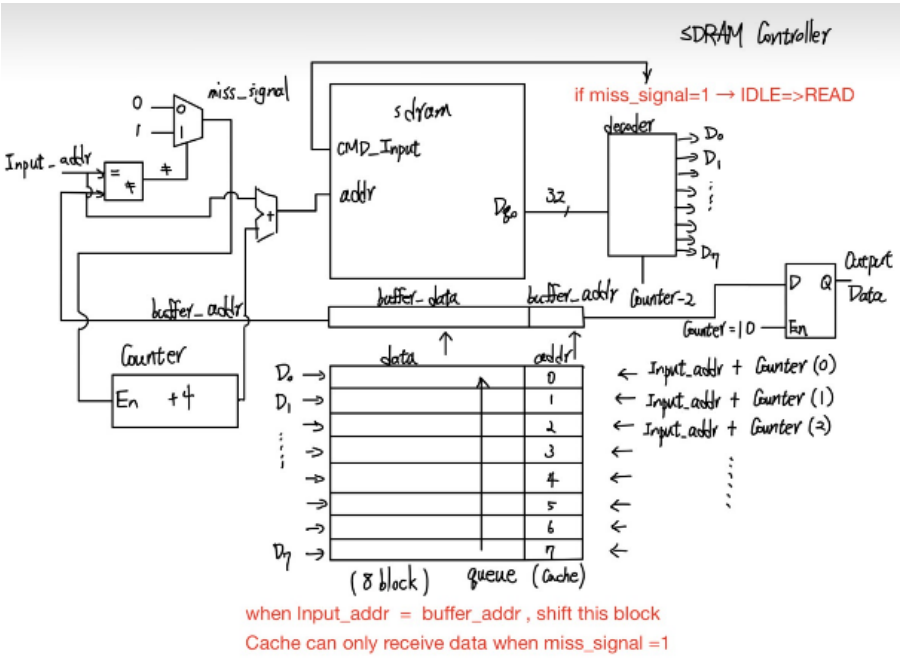


依照該 LAB 給的 SDRAM 運作順序，可以看到大多為 IDLE -> READ -> READ_RES ->IDLE 佔據大部分時間(8 筆資料總共 64~68cycle)，因此我們在其中加入 cache 並於 READ 狀態時，先 prefetch 資料最後再輸出，如下圖。



經由 prefetch 後再輸出資料，省去了每讀取一次資料都要再經過 READ_RES 的時間，將 8 筆資料輸出的時間減少至 54~58cycle。

二、 Prefetch Schematic：



我們設計是當 CPU 需要 Sdram 中相對應 addr 的資料時，一率從 buffer 中讀取，並透過 cache 將所需資料先 prefetch 下來，利用 shift 的方式推送至 buffer 中(該設計僅針對 Read 部分，Write 則直接寫入 sdram)。

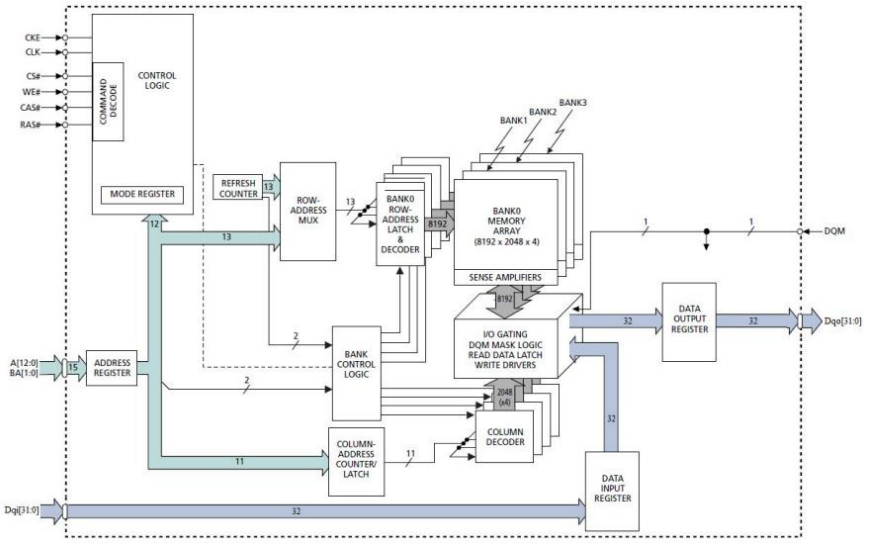
當要求的 addr 進入時，與 buffer 中的 Addr 進行比對，相同則代表 buffer 中資料為所需，即可直接輸出，省去每次都要重新讀取一次 Sdram 的過程。

發生 addr 比對不相等，則輸出 miss_signal 訊號，啟動 counter 與 cache 的接收訊號，並不斷讀取 Sdram 直到 cache 內資料全部替換為新的。

觀察到 CPU 可連續輸出 8 筆 addr(該部分仍在研究)來要求資料，故選用 8 block cache 來實現該電路。

因每段指令都以+4 為主要，在 cache 中 prefetch 的 addr 即以+4 為主，存入 8 筆。

三、 Bank Interleave for Code and Data：



1. Data：

```
38200000 <A>:
38200000:      0000                .2byte  0x0
38200002:      0000                .2byte  0x0
38200004:      0001                .2byte  0x1
38200006:      0000                .2byte  0x0
38200008:      0002                .2byte  0x2
3820000a:      0000                .2byte  0x0
3820000c:      00000003            lb      zero,0(zero) # 0 <__DYNAMIC>
38200010:      0000                .2byte  0x0
38200012:      0000                .2byte  0x0
38200014:      0001                .2byte  0x1
38200016:      0000                .2byte  0x0
38200018:      0002                .2byte  0x2
3820001a:      0000                .2byte  0x0
3820001c:      00000003            lb      zero,0(zero) # 0 <__DYNAMIC>
38200020:      0000                .2byte  0x0
38200022:      0000                .2byte  0x0
38200024:      0001                .2byte  0x1
38200026:      0000                .2byte  0x0
38200028:      0002                .2byte  0x2
3820002a:      0000                .2byte  0x0
3820002c:      00000003            lb      zero,0(zero) # 0 <__DYNAMIC>
38200030:      0000                .2byte  0x0
38200032:      0000                .2byte  0x0
38200034:      0001                .2byte  0x1
38200036:      0000                .2byte  0x0
38200038:      0002                .2byte  0x2
3820003a:      0000                .2byte  0x0
3820003c:      00000003            lb      zero,0(zero) # 0 <__DYNAMIC>

38200040 <B>:
38200040:      0001                .2byte  0x1
38200042:      0000                .2byte  0x0
38200044:      0002                .2byte  0x2
38200046:      0000                .2byte  0x0
38200048:      00000003            lb      zero,0(zero) # 0 <__DYNAMIC>
3820004c:      0004                .2byte  0x4
3820004e:      0000                .2byte  0x0
38200050:      0005                .2byte  0x5
38200052:      0000                .2byte  0x0
38200054:      0006                .2byte  0x6
38200056:      0000                .2byte  0x0
38200058:      00000007            .4byte  0x7
3820005c:      0008                .2byte  0x8
3820005e:      0000                .2byte  0x0
38200060:      0009                .2byte  0x9
38200062:      0000                .2byte  0x0
38200064:      000a                .2byte  0xa
38200066:      0000                .2byte  0x0
38200068:      0000000b            .4byte  0xb
3820006c:      000c                .2byte  0xc
3820006e:      0000                .2byte  0x0
38200070:      000d                .2byte  0xd
38200072:      0000                .2byte  0x0
38200074:      000e                .2byte  0xe
38200076:      0000                .2byte  0x0
38200078:      0000000f            fence   unknown,unknown
3820007c:      0010                .2byte  0x10
```

2. Code：

```
38000000 <matmul>:
38000000:       fd010113                addi    sp,sp,-48
38000004:       02112623                sw      ra,44(sp)
38000008:       02812423                sw      s0,40(sp)
3800000c:       03010413                addi    s0,sp,48
38000010:       fe042623                sw      zero,-20(s0)
38000014:       fc042e23                sw      zero,-36(s0)
38000018:       fe042623                sw      zero,-20(s0)
3800001c:       0e80006f                j       38000104 <matmul+0x104>
38000020:       fe042423                sw      zero,-24(s0)
38000024:       0c80006f                j       380000ec <matmul+0xec>
38000028:       fe042023                sw      zero,-32(s0)
3800002c:       fe042223                sw      zero,-28(s0)
38000030:       07c0006f                j       380000ac <matmul+0xac>
38000034:       fec42783                lw      a5,-20(s0)
38000038:       00279713                slli    a4,a5,0x2
3800003c:       fe442783                lw      a5,-28(s0)
38000040:       00f707b3                add     a5,a4,a5
38000044:       38200737                lui     a4,0x38200
```

```
3800019c:       0007a783                lw      a5,0(a5)
380001a0:       01079713                slli    a4,a5,0x10
380001a4:       260007b7                lui     a5,0x26000
380001a8:       00c78793                addi    a5,a5,12 # 2600000c <_esram_rom+0x15fff8bc>
380001ac:       00e7a023                sw      a4,0(a5)
380001b0:       fec42783                lw      a5,-20(s0)
380001b4:       02478793                addi    a5,a5,36
380001b8:       0007a783                lw      a5,0(a5)
380001bc:       01079713                slli    a4,a5,0x10
380001c0:       260007b7                lui     a5,0x26000
380001c4:       00c78793                addi    a5,a5,12 # 2600000c <_esram_rom+0x15fff8bc>
380001c8:       00e7a023                sw      a4,0(a5)
380001cc:       00000013                nop
380001d0:       00078513                mv      a0,a5
380001d4:       01c12083                lw      ra,28(sp)
380001d8:       01812403                lw      s0,24(sp)
380001dc:       02010113                addi    sp,sp,32
380001e0:       00008067                ret
```

四、 How to Modify the Linker to Load Address/Data in Two different bank

```
MEMORY {
        vexriscv_debug : ORIGIN = 0xf00f0000, LENGTH = 0x00000100
        /*dff : ORIGIN = 0x00000000, LENGTH = 0x00000400*/
        dff : ORIGIN = 0x38000400, LENGTH = 0x000003FF
        dff2 : ORIGIN = 0x00000400, LENGTH = 0x00000200
        flash : ORIGIN = 0x10000000, LENGTH = 0x01000000
        mprj : ORIGIN = 0x30000000, LENGTH = 0x00100000
        mprjram : ORIGIN = 0x38000000, LENGTH = 0x000003FF
        hk : ORIGIN = 0x26000000, LENGTH = 0x00100000
        csr : ORIGIN = 0xf0000000, LENGTH = 0x00010000
}
```

在 sections.lds 中更改 dff 位址，以及將 marjram 原先的 LENGTH 調低，分出空間給 data。

```
assign Mapped_BA = user_addr[11:10];
assign Mapped_CA = user_addr[9:2];
assign addr = {13'd0, Mapped_BA, Mapped_CA};
```

因每次的位址皆為加 4，這邊自動將其右移 2bit，並利用第 10、11bit 當作 bank 的切換。