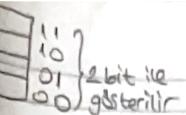


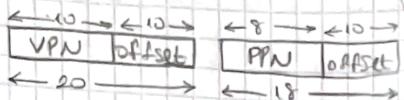
$2^2$  byte



ÖR: Page size  $1KB \Rightarrow 2^{10}$  byte

Veri yolu 32 bit  $\Rightarrow 1$  word

VA 20 bit PA 18 bit



①  $\log_2^{2^10} = 10$  bit = offset değeri

a)  $0x4AA4 = 0000\ 0100\ 1010\ 1010\ 0100$   
VPN   offset  
(N=1) TCB'ye nüf.  $0x012 = \text{VPN}$ ,  $0x2A4 = \text{offset}$

PPN = 0xA1      offset = 0x2A4

$\frac{10}{2}\ 10\ 0001\ 10\ 10\ 10\ 0100$   
A        1    2    A        4    4

PA = 0x286A4

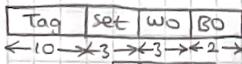
b) 0x078E6

(V=0, page table'dan al)

ÖR: 8-way assoc. cache, PA 18 bit

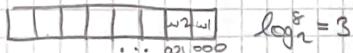
block size 8 word, 1 word 32 bit

cache size 512 word



① B0  $\rightarrow [B4\ B3\ B2\ B1]$  1 word  $= \log_2^4 = 2$

② W0  $\rightarrow$  Block'taki word'u adreslemek



③ set  $\rightarrow$  8 adet yan yan cache

satır =  $\frac{512 \text{ word}}{64 \text{ word}} = 8 \text{ satır}$   $\log_2^8 = 3$

Critical path: En uzun süren işlevi clock bulusuna göre yapılır. (single-cycle)  
 \* Hatırda instruction memory ve data memory single-cycle da aynı yerde ama gerekli hizatta birlikte oluyor.  
 \* Single-cycle da 3 farklı toplayıcı var.

### Multicycle处理器

\* her bir clock ile değil, görevi parçalı olarak her bir clock ile işlem yapılır.

- 1-Fetch  $\rightarrow$  instruction memory'den komutu okur.
- 2-Decode  $\rightarrow$  gelen kodun ne işteğini anlar.
- 3-RS'yi oku
- 4-ALU  $\rightarrow$  rs+imm değerini bulur.
- 5-Adresi rs+imm olacak şekilde oku
- 6-Orduğunu değerini rd'ye aktarır.

### 14. hafta

\* Cache'de veri bulunamadığı zaman RAM'den, RAM'de de yoksa hard disk'ten veri çekilir.

\* RAM'den cache'e varlıklar bloklar halinde alınır. Hard diskten cache'e page şeklinde alınır.

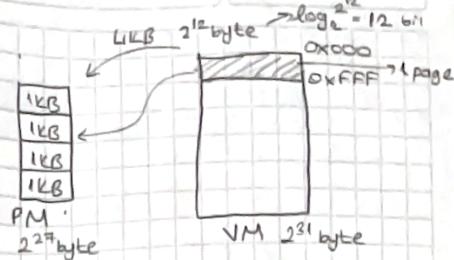
NOT: virtual memory'de program kosturulurken kayalı ve büyük bir hafıza alanı ayrırlar.

\* Sanal adreslerde program kosturulur.

virtual address mapping

\* page size ter seferde RAM'e gönülacak bilgi metindir.

VM size  $\rightarrow$  2GB  
 PM "  $\rightarrow$  128MB  
 page "  $\rightarrow$  4KB

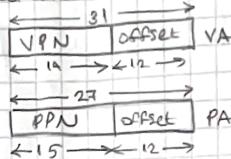


VPN | Page offset 31 bit

  ↑  
 translation 112  
 ↓  
 PPN | Page offset 27 bit

\* VM address bitti  $>$  PM address bitti dikkat ettiğimde

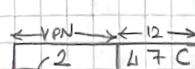
zorunda



ÖR: 0x247C VA ise PA = ?

Page table	V
0.	0
0.	0
UXFFFF	1
UXFFFF	1

verilmesi gereklidir.



0x247C'den katı direkt yerleştir.

0x247C  
 ↓  
 VPN      12 →  
 ↓  
 2      47 C  
 ↓  
 2 numaralı satırı baktır.

2 numaralı satırı baktır.

0x7FFF 47 C      0x7FFF 47 C

ÖR: 0x5F20  
 ↓  
 VPN      offset  
 ↓  
 5. satırındaki PPN değeri 5F20'nin kesişme eklenir.

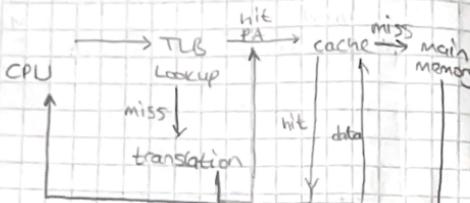
NOT: valid değilse diskten çekilmeli sil gerekir. (page fault)

Dezavantajlar:

- Page table is large
- Load/store requires 2 main memory accesses  
↳ translation ve access data
- cuts memory performance in half

Gov. girişli hale getirilerek iyileştirilebilir.

lw \$5, L(\$w) # \$5 < M[5444]



ÖR: VPN 20 bit, PPN 12 bit, page offset 12 bit  
VA'ları varilen PA'ları bul.

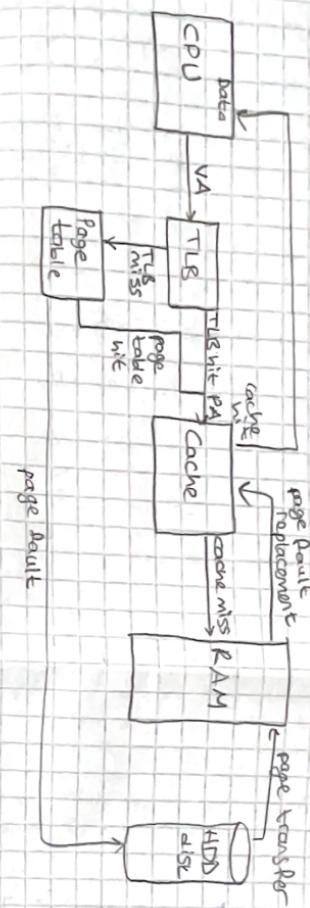
V D TAG(VPN) Data(PPN)

1	1	01AF4	FFF
0	0	0E45F	E03
0	0	012FF	2FO
1	0	01A37	788
1	0	02BB4	45C
0	1	03CA0	657

a) 02BB41A65  
45CA65

c) 0D34E91DC  
miss, page tablosuna  
bağılmalı.

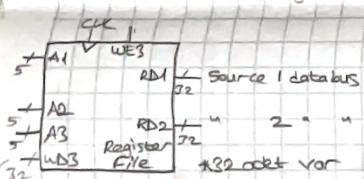
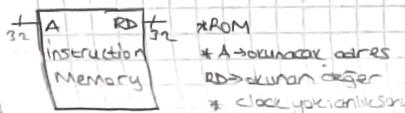
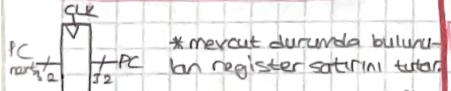
b) 0E45 F832  
valid değil page tablo-  
sunu核算malı,  
d) 03CA0777  
miss, page tablosuna  
bağılmalı.



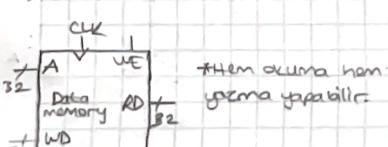
### RISC-V subset instructions

R-type ALU  
 • add/sub  
 • and/or/slt  
 Memory  
 • lw/sw  
 Branch  
 • beq

### RISC-V architectural elements



A1 A2 A3  
 \* aynı anda kullanılır.  
 source → destination sayesinde  
 destination databus



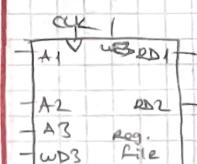
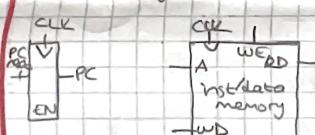
### SINGLE CYCLE

- \* simple
- \* kritik yol cycle süresini etkiler. (lw)
- \* separate memories for inst. and data
- \* 3 adders/ALUs

### MULTI-CYCLE

- \* shorter inst. take fewer steps
- \* can re-use hardware
- \* run faster
- \* reuse expensive hardware on multiple cycles

\* first data path ve control  
lisiinde de aynı



### MEMORY

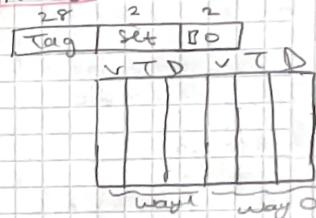
#### Cache Terminoloji

- \* Capacity: number of data bytes in cache
- \* Block size: bytes of data brought into cache at once

#### Cache

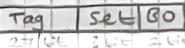
- direct mapped
- N-way ass., fully ass.
- n blocks per set
- 1 blocks per set

### N-way

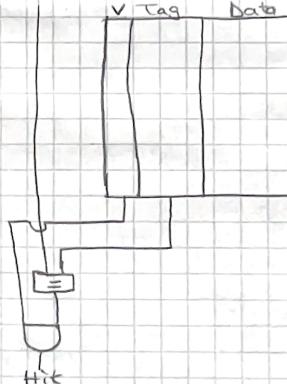


\* larger caches have lower miss rates, longer access times

### Direct mapped:



27 bit 3 bit 2 bit



## DDC-Cn6

- \* RISC-V has 32 32-bit reg.
- \* Reg. faster than memory.

$a = b - c$      $\#s0 = a, s1 = b, s2 = c$   
 $\text{sub } a, b, c$

$a = b + c$      $\text{add } a, b, c$

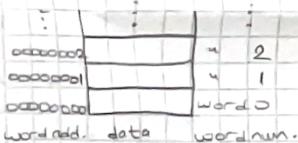
$a = b + b$      $\text{addi } a, b, b$

### Memory

↓  
word-add.    byte addressable  
memory    memory

### Word add. memory

- \* word adresleri 4'er artar



\* memory read / load  
 $lw$  destination offset(base)

bitler döşenir. ikişini toplanır.

\* memory write store  
 $sw$  dest. offset(base)

Byte add. memory  
\* word adresleri 4'er artar



\* RISC-V is byte-addressed.  
\* LW ve SW komut yazımı ayndır. Tek fark koducu word olduğundan offset'i 4'e bölererek buluruz.

### Constants

↓                          ↓  
12 bit                    32 bit  
\* 12 bit ve altıinda kullanılır.    \* 32 bit kullanılır.

lli: ilk 20 biti register'a kaydırır.  
kalan düşük 12 biti 0'dır. kalan  
12 addi ile eklenir.

NOT: addi sign-extends youn  
yani 12 bitten önceki 20  
biti sayının son gitiliye  
doldurur. (ignoret biti)

and: masking   or: combining  
xor: inverting

### Shift

sll      .sra (shift left msb tekrar eder)  
.srl  
\* shift amount is in (lower 5  
bits of) a register

timm shift 0-31 arası

### Branching

↓  
conditional                    unconditional  
.beq v6                    .jump v6

### Function calls

#### caller:

- \* passes arguments to callee
- \* jumps to callee

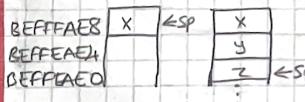
#### callee:

- \* performs the func.
- \* returns result to caller
- \* " to point of call
- \* must not overwrite registers or memory needed by caller

call func.: jump and link (.jal)  
return from func.: jump reg (.jr ra)  
arguments: a0-a7  
return value: a0

### Stack

\* SP (stack pointer) points to  
top of the stack  
\* grows down (from higher to lower)



Mult-cycle osman

1 or:

1-instruction (IM):

\*IM'den ins. okun  
nur ve PC'inin son  
razi deger hesap  
lanır.

2-Register (RF):

\*Ins. decode edilir.  
\*Reg. file'dan oku  
nacak reg. adresini  
belirlenir.

\*Beq döma int. lok  
ALLout reg. 4'tür  
PC + offset yapılır.

3-Execution, mem.  
bez-cmp:  
\*decode edilmiş  
işleme göre devam  
ediliyor.

NOT: Bu aranada  
eger beq  
inst. ise  
istem tamam  
lamış olur.  
Daha önce ALLU  
out yazılıdığı  
için geni lis. gel  
meye hazırda.

H-DM :

- \*Memory access
- or R-Type

NOT: R-Type ve SW

Komutları,

buradan top-

ka H-clock

dece lw kd-

mutu S-clock

sinyalini

ister.

Single:

- Fetch
- Decode
- Execute
- Memory
- Writeback

NOTLAR

\*Pipeline ile tekn.  
tamamlanma süresi atal  
maz, sistemler toplan  
perf. artar.

Structural Hazard:

bu H-clock

ile biter. San- support combination

dece lw kd-

mutu S-clock

Data Hazard:

#Bir önceki veri in-

ister.

boska yerde kullanıl-

ması!

Control Hazard:

#beq ?