

* 16-bit MIPS * "Soft" Team

Inst. \leftarrow APP \leftarrow ماینریتے کو دیکھوں گا اسکے لئے نہیں
→ Inst. \leftarrow کتبے کو دیکھوں گا اسکے لئے نہیں

لیکن اسکے لئے Inst. \leftarrow ماینریتے کا نہیں فناوری:

→ R-TYPE:

Registers \leftarrow میکنے میں وہ مفہوم تھا کہ جو اس کے مطابق

ویخواہ

→ I-TYPE: Reg. \leftarrow میکنے میں وہ مفہوم تھا کہ جو اس کے مطابق Immediate, Registers \leftarrow Data Memory \leftarrow Cond. JUMP. اور

→ J-TYPE: جو اس کے مطابق UNCond. - JUMP

16-bit \leftarrow میکنے کے لئے دلوقتی 16 بتے دیں اسکے لئے Inst. \leftarrow دو ممکنہ طور پر اس کے لئے 16-bit

Memory \leftarrow 0 to *

16 bit \leftarrow address \rightarrow access

$2^{16} \times 16$ \leftarrow size \leftarrow 16

\leftarrow Memory \leftarrow 0 to

16 \leftarrow کل مجموعہ \leftarrow 2^{16}

16-bit

انڑی Inst. \leftarrow تقسیم کے لئے تھا کہ نشوونے کے لئے میکنے کے لئے نہیں فناوری: TYPES \leftarrow no TYPE \leftarrow Inst. \leftarrow

→ R-TYPE:

4bit 3-bit 3-bit 3-bit func.

OP-Code	Rs	RT	RD	3-bit
---------	----	----	----	-------

يعبروا عن 4bit أول بوك up
عن R-TYPE في OP-Code
R-TYPE: INST لـ الـ
OR, AND, SUB, ADD لـ الـ
OUT, IN, JR, SRL, SLL, XOR, NOR
لـ الـ OP-Code
func الـ 3bit طبقاً لـ المـ
func الـ 8 زرورـاً لـ 3bit لـ
ـ الـ INST الـ R-TYPE الـ

وفـ الـ address الـ INST الـ
الـ Reg-Access الـ 3bit
ـ RegS الـ INST فالـ Reg-Access
ـ الـ address الـ

→ I-TYPE:

OP-Code Rs RT imm.

4-bit	3-bit	3-bit	6-bit
-------	-------	-------	-------

ـ OP-Code لـ 16
ـ INST لـ 16
ـ I-TYPE لـ 16
ـ INST لـ

LW, ORI, ANDI و ADDI
BEQ, BNQ, PW

• address ١١ و ٦ bits
first ٣ bits لـ value Reg. ٣ bits
→ INST ١١ و Regs

• LW بـ Immediate و ٦ bits

address ١١ و value ٦ bits
INST ١١ و ٦ bits لـ value ٦ bits
• جـ INST ١١ و ٦ bits

* J-type =

OP-Code	Target
4-bit	12-bit

INST ١١ و ٦ bits لـ OP-Code
• J-type ١١ و ٦ bits

J, JAL, JR, BE

• to inst as Target ١١ و ٦ bits
• ٦ bits address

٥٨ instructions INST
Logs into family

NOP Halt
• ٤ bits stop ٤ bits

الثقافة النبوية.

لما كان على المبرمجين
عزم اثراً في الـ Hardware

H.W. يحترف كل شيء

Inst. كل شيء يحترف كل

Inst. كل شيء يحترف كل



* ADD Rd, Rs, Rt

Harvard الـ و MIPS الـ

Data الـ واحد Two Mem. Line

Inst. الـ فيها الـ واحد
counter الـ معاشرة P.C Line

Inst. الـ لها address الـ خطوة

Reg. file و ALU

الـ الـ Control unit

fetch الـ Inst. في الـ

P.C في الـ address الـ دخول

one word one Inst. Memory

address الـ فيها الـ P.C

الـ no Inst. بدلها رعن

. one word bus Memory

* Fetch: $\text{INST MEM} \leftarrow \text{PG}$ ina PG

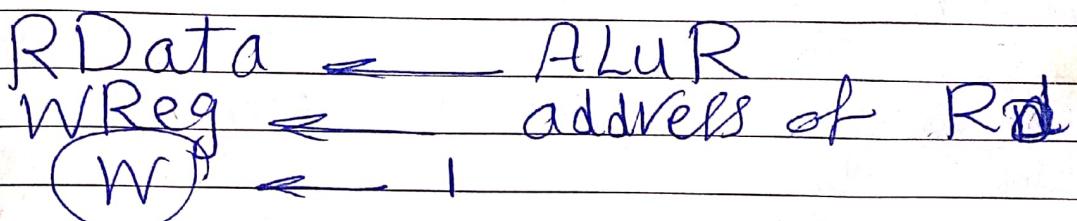
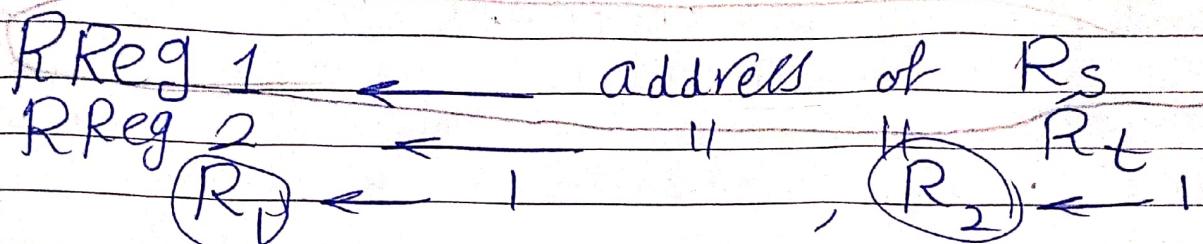
Control unit will op code $11 \rightarrow 6$.

① R-type:

* ADD R_d, R_s, R_t

$$\rightarrow R_d \leftarrow R_s + R_t \text{ (Macro Inst)}$$

Micro Inst:



* SUB $R_d, R_s, R_t =$

$$\rightarrow R_d \leftarrow R_s - R_t \text{ (Macro Inst)}$$

Micro Inst:

$R\text{Reg}_1 = \text{address of } R_s$

$R\text{Reg}_2 \leftarrow \text{II} \quad R_t$

$ALUA \leftarrow RP_1$, $ALUB \leftarrow RP_2$
 (SUB) \leftarrow « التعبير الفنى »

$RData \leftarrow ALUR$
 $wReg \leftarrow$ address of Rd
 $(W) \leftarrow 1$

* AND Rd, Rs, Rt

$\rightarrow Rd \leftarrow Rs$ and Rt (Macro Inst)

$RReg_1 \leftarrow$ address of Rs

$RReg_1 \leftarrow$ " " " " Rt

$(R_1) \leftarrow 1$, $(R_2) \leftarrow 1$

$ALUA \leftarrow RD_1$, $ALUB \leftarrow RD_2$

$(AND) \leftarrow 1$

$RData \leftarrow ALUR$

$wReg \leftarrow$ address of Rd

$(W) \leftarrow 1$

* OR Rd, Rs, Rt

The same as AND

but exchange $(OR) \leftarrow 1$

* NOR Rd, Rs, Rt

The same as AND
but exchange $(NOR) \leftarrow 1$

* XOR Rd, Rs, Rt

The same but
 $\text{XOR} \leftarrow 1$

* SLL Rd, Rs

Rd \leftarrow Shift Left (Rs)

RReg 1 \leftarrow address of Rs

$\text{R}_1 \leftarrow 1$

ALU ~~RData~~ \leftarrow RD₁

ALUA \leftarrow RD₁
 $\text{SLL} \leftarrow 1$

غير مصرح بالرسم في هذه الصفحة

RData \leftarrow ALUR

WReg \leftarrow address of RD

$W \leftarrow 1$

* SRL Rd, Rs

Rd \leftarrow Shift Right (Rs)

The Same as SLL

but $SRL \leftarrow 1$

* JR

العنوان المدخل $J1 \leftarrow PG$ فار
عنوان المخرج $J2 \leftarrow RS$ مدخل

RReg 1 \leftarrow address of RS

(R1) \leftarrow 1

PG $\leftarrow RD_1$

* IN Rd

Input \leftarrow Input Port $J1 \leftarrow$ no
Rd $J1 \rightarrow$ yes

WReg 1 \leftarrow address of Rd

RData \leftarrow from input Port

(W1) \leftarrow 1

* OUT Rs

welb, Rs no Data \rightarrow
output Port $J1 \leftarrow$ dc

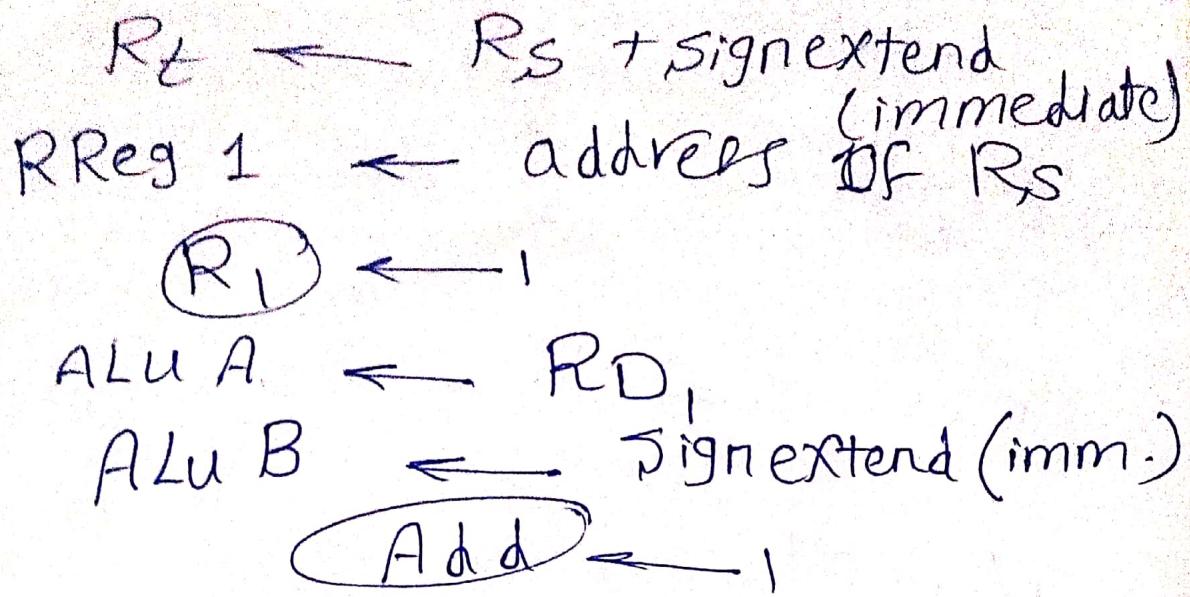
RReg 1 \leftarrow address of Rs

(R1) \leftarrow 1

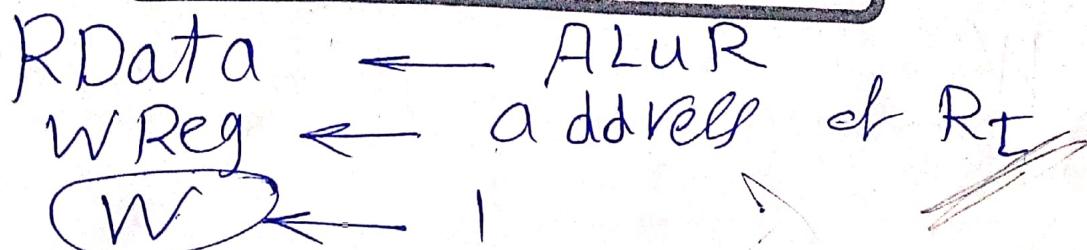
Output Port $\leftarrow RD_1$

② I-TYPE:

* ADDi Rt, Rs, immediate



غير مصرح بالرسم في هذه الصفحة



* ANDi

The same but



* ORi



* LW Rt, offset(Rs)

Rt \leftarrow M[Rs + sign extend(imm)]
العنوان الفيزيائي = address of Rs

RReg 1

(R1)

\leftarrow

ALUA \leftarrow RD1

ALUB \leftarrow sign extend (imm)

(ADD)

\leftarrow

~~MData~~ \leftarrow ALU

Madd \leftarrow ALUR

Memory
address

Memory
out

RData \leftarrow Mout

WReg \leftarrow address of Rt

(W)

~~W~~ \leftarrow

* SW Rt, offset (Rs)

M[Rs + sign extend (imm)] \leftarrow Rt

RReg 1 \leftarrow address of Rs

(R1)

ALUA \leftarrow RD1

ALUB \leftarrow sign extend (imm)

(ADD)

RReg2 ← address of Rt
 $(R_2) \leftarrow 1$

MIN ← RP₂
MAdd ← ALUR
 $M\text{Write} \leftarrow 1$

* BEQ : Rs , Rt , offset

IF Rs = Rt

PG ← PG + sign extend (imm)

غير مصرح بالرسم في هذه الصفحة

RReg1 ← address of Rs

RReg2 ← " " " Rt

$(R_1) \leftarrow 1$, $(R_2) \leftarrow 1$

ALUA ← RD₁ , ALUB ← RP₂

$(SUB) \leftarrow 1$

IF zero flag ← 1

~~Add1~~
~~ALUA~~ ← PG

~~Add2~~ ALUB ← sign extend (imm)
 $(ADD) \leftarrow 1$ PC ← ALUR
addres sum 1

* BNQ:

The same but الاشتغالية

IF zeroFlag = 0

③ J-TYPE:

* J-Target

$PG \leftarrow PC \parallel Target \parallel 0$

↓

15:13
3bit 12 bit 0

* JAL Target

فے رقم Reg \rightarrow
ار بتابع address \parallel Regfile \parallel
ایجاتی Inst

فے رقم کو لیتے سے ممکنہ تھا
کہ address کو فرمائیں اور
وید کو کہا جائے
PC \rightarrow کہا جائے

$PG \leftarrow PC \parallel Target \parallel 0$

15:13

④ No types Inst

* NOP: CU \parallel تردد IC

فیکھوں کے مکانیک OP-Code ہے

* Halt: PG \rightarrow خر ہے
سچی Inst اس کے بعد address

- for ALU: مقدار دخول ALU من ALU
 - ADD (ناتج يدخل ALU من مدخل الجمع) و ناتج من
 - ALU then FG \rightarrow 0
- NOR و XOR و OR و AND مدخل logic \rightarrow ②
- SRL و SLL دخل shift دار \rightarrow ③
- دخل Result دار منها ما دار
 عاشر آخر منها دوك و مدخل تحكم
 ↗ Operation دو دوك لـ zero flag دار
 تعرفه bit نتائجهما 0.
- Operations دخلها 2 inputs ديلها \rightarrow
 ↗ دخلها 16-bit كل واحد دوك
 ↗ دخلها 3-bit دوك operation input دار
 ↗ دخلها 16-bit دوك result output دار
 ↗ دخلها 1 bit دوك output دار
 ↗ دخلها 1 bit دوك zero flag دار

ALU دار دو دوك control signals دار
 دوك ALU control دوك CU دار دوك
 دوك هو عبارة عن 3-bit

→ for Regfile: عاشر دخل Regfile دوك
 8 Regs دوك دوك Regfile دوك
 دوك دوك 16-bit دوك Regs كل دوك
 دوك دوك اقوى دوك دوك
 دوك دوك في نفس الوقت وبالتساوي

3-bit \leftarrow input ٢ inputs متطلبات
accept لـ address ادخل عليهم

✓ RReg2, RReg1 وهما two Regs لـ ومتطلبات منهم من Read لـ value
RReg1 \rightarrow R₁ وهذا signal ادعى
· RReg2 \rightarrow R₂ و

وبالتالي Reg متطلبات في
فيه 3-bit معرفة input
Reg لـ address ادخل عليه او
WReg وهو الذي عاشر الكتاب في
Data لـ input ومتطلبات دخل عليه او
اللى هكتها فار Reg ده

غير مصرح بكتابه أية إجابات في هذه الصفحة

وبالتالي في RData او الى هيكون
16-bit
متطلبات بمدخل Signal اقول دلته
1 bit W وهى هيكون

Regs \rightarrow output بالمعنى
هيكون Data او binary

R_{D2}, R_{D1}

Control Signals \leftarrow Reg. file او متطلبات

W R₁, R₂*

→ for Inst & Data Memory:

* Instruction Memory

Memory 2^{16} فتحة
وراء كل مدخل 16 bit
(design)
العنوان address لـ input خلص
مع INST في المخرج 16 bit
 \Rightarrow Inst لـ output و يدخل
و الباقي طولهم 16-bit

* Data Memory:

2^{16} فتحة Memory متلاع بـ خروج
16 bit وكل مدخل و كل مدخل
(design)

عنوان address لـ input اقدرها
McMAddress لـ output يدخل
عنوان address لـ input

ولو عاتر عنوان address
عنوان address لـ input

وحدة الـ data لـ input
· INMEM لـ input

- MEMW و "الـ" signals

الخطوة الأولى هي إدخال output إلى
Memory للوصول

الخطوة الثانية هي إدخال Control Signals إلى
Memory → 1 bit

→ for IR:

الخطوة الثالثة هي إدخال INSTRUCTION إلى

Instruction Memory إلى Inst. Register
الخطوة الرابعة هي إدخال Instruction من Inst. Register
إلى Inst. Decoder
الخطوة الخامسة هي إدخال bits إلى Decoder
الخطوة السادسة هي إدخال CU_N op. Code إلى Decoder
الخطوة السابعة هي إدخال bits إلى Decoder و بحث عن

16-bit address إلى Reg أو

→ for PG:

الخطوة الخامسة هي إدخال address إلى Reg أو
الخطوة السادسة هي إدخال Next Inst إلى
الخطوة السابعة هي إدخال address إلى
الخطوة الثامنة هي إدخال Source

input إلى output
Instruction Memory
16-bit address

مقدمة في الميكانيكا

→ for Control unit:

- IR || no OP Code || ~~لیست~~
- ~~فایر~~ INST- || ~~کولوپ~~ ~~لیست~~
- ~~فایر~~ OPatio Control Signals ~~لیست~~
- ~~فایر~~ Signals || ~~لیست~~

→ for signed extend:

6 bit ~~لیست~~ Block no ~~لیست~~
~~لیست~~, ~~لیست~~ no extend ~~لیست~~
Sign bit ~~لیست~~