Control inst. → Jump-branch; farklı yere gidiyor.

inst	Syntax	cond	branch	link ad
balrz	balz $rs, $rd	if Status[Z]=1	PC←R[rs]	R[rd]←PC+4

balrz ↓ it checks the status zero flag → branch and link

(Status[Z]) ↓ aritmetik result of previous instruction

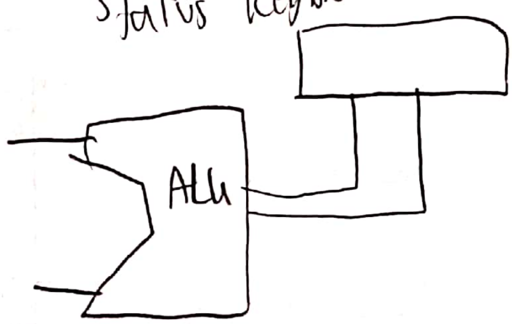
We are using rs to update PC if cond. true. [if zero flag = 1]

---

balrn   balrn $rs, $rd   if Status[N]=1   P— — — —

balrn ↓ branch and link register negative

if it's ? rs to PC.



---

balz ↓

branch and link zero → we don't have any register content. we have target in jump we don't have any condition. If one of previous inst. resulted zero branch. → jump unconditional'dır

---

bgez → branch greater or equal to 0

Branch address → beq ile aynı ama burda condition var → if R[rs]>=0

bgez içinde L olmadığı için link yapılmaz.

jmor → jump memory or, we get aritmetic result of or

---

jalr → no condition → update PC with rs.] jump and link register
store PC+4 in rd register.

---

jrs → fetch single content. Memory content will set in PC.
single register as an address

---

jal - jump and link
↳ L → stands for saving our return adress to R[31]

---

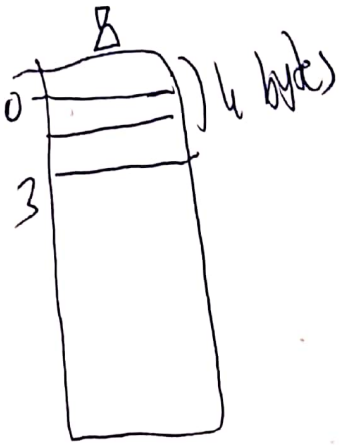
sllv → use different register for shift amount

Need to write truth table for j/b control unit
more branch & jump signals

jmadd → jump memory add

kod 78-85 arası 8 mux var. → bu artırılıp azaltılabilir.

~~Data memory~~ ~~47-52 write byte~~



121-124 → reads th
126 - debug