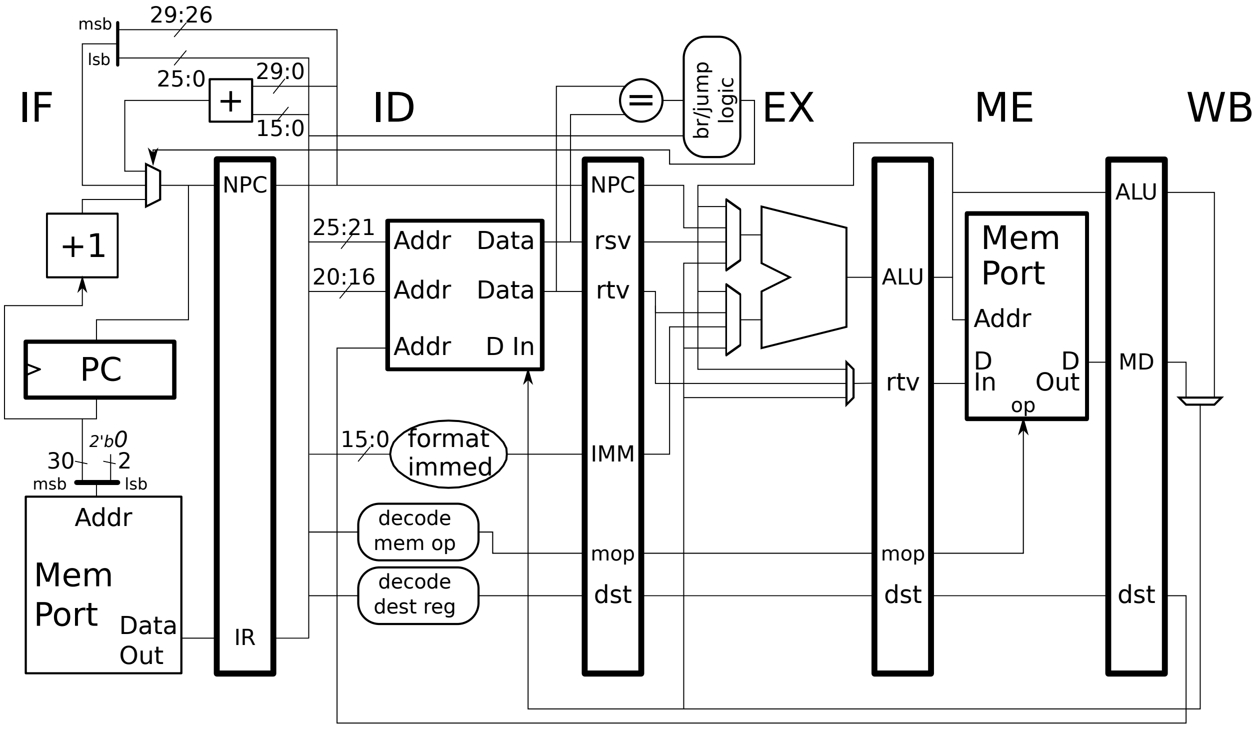
LSU EE 4720 Homework 3 Due: 24 February 2017

To help in solving this problem it might be useful to study the solutions to the following problems which involve hardware implementing branches in the statically scheduled five-stage MIPS implementation we've been working with: Spring 2016 Homework 3 (SPARC-like branch instruction), Spring 2015 Homework 2 Problem 2 (use reg bits for larger displacement) and Problem 3 (logic for IF-stage mux), Spring 2015 Homework 3 Problem 2 (implement bgt, but resolve it in EX), Spring 2011 Final Exam Problem 1 (resolve in ME, with bypass).

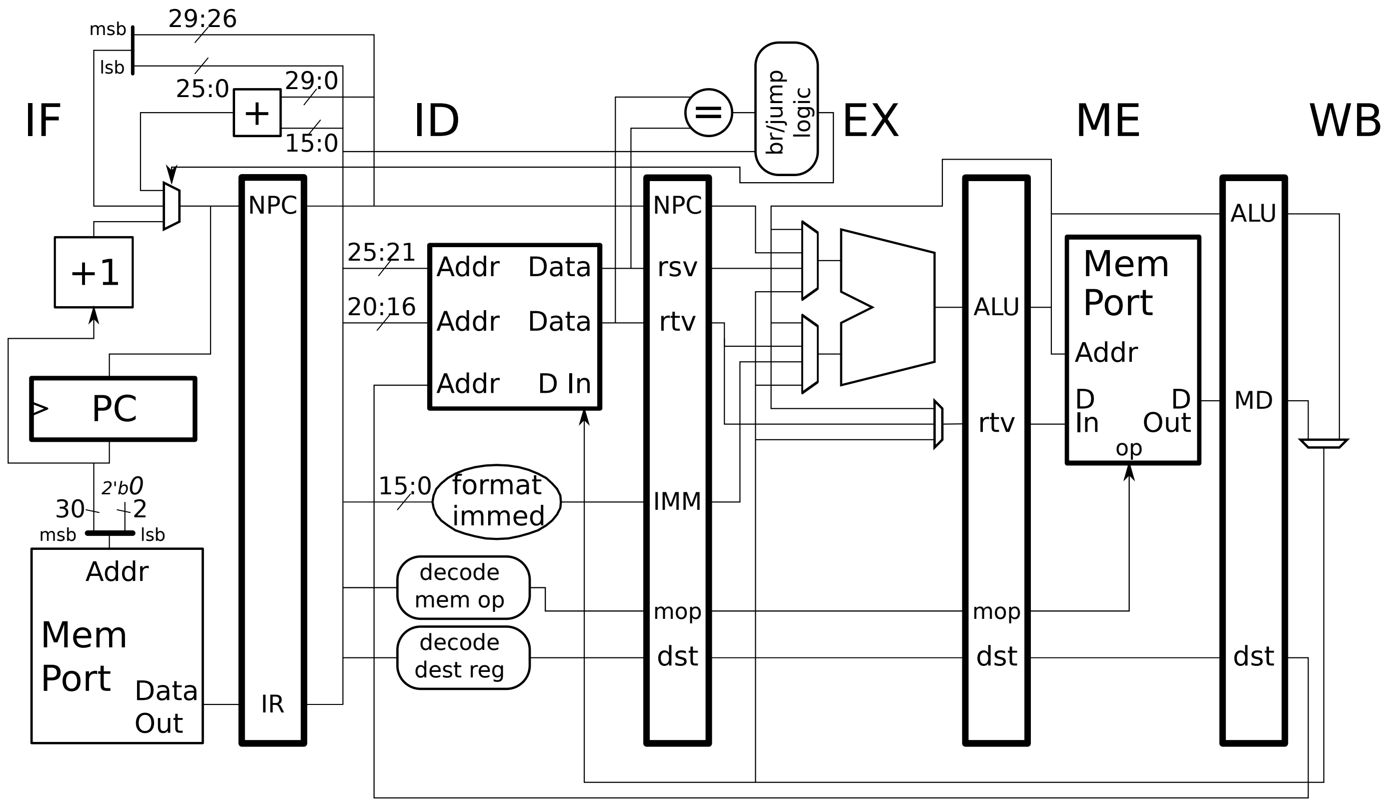
Problem 1: Modify the implementation below so that it implements the MIPS II `bgezall` instruction, see the subproblems for details on the hardware to be designed. See the MIPS ISA documentation linked to the course Web page for a description of the `bgezall` instruction. An Inkscape SVG version of the illustration below can be found at `http://www.ece.lsu.edu/ee4720/2017/hw03-p1.svg`. The illustration also appears on the next page.



(*a*) Design control logic to detect the instruction and connect its output to the $\boxed{\text{br/jump logic}}$ cloud. The control logic should consist of basic gates, **not** a box like $\boxed{\texttt{bgezall}}$.

(*b*) Design the control logic to squash the delay slot instruction when `bgezall` is not taken. The control logic should squash the delay slot instruction by changing its destination register and memory operation. Be sure that the control logic squashes the correct instruction, and does so only when `bgezall` is not taken. Do not rely on magic clouds [tm].

(*c*) Add datapath or make other changes needed to compute the return address. Note that `NPC` is already connected to the ALU. Consider inexpensive ways to compute the second operand. (Adding a 32-bit `ID/EX` pipeline latch is not considered inexpensive for this problem.)


IF
ID
EX
ME
WB
msb
lsb
29:26
25:0
29:0
15:0
+
+1
PC
2'b0
30
2
msb
lsb
Addr
Mem Port
Data Out
NPC
IR
25:21
20:16
Addr
Data
Addr
Data
Addr
D In
15:0
format immed
decode mem op
decode dest reg
br/jump logic
=
NPC
rsv
rtv
IMM
mop
dst
ALU
rtv
mop
dst
Mem Port
Addr
D In
D Out
op
ALU
MD
dst