**Extra Credit Optimizations –** Team 10

Matthew Bachmeier, Amanda Becker, Garret Huibregtse

**3. Replace RCA with CLA (0 –0.5 points):** We replaced the rippler-carry adder used anywhere in the processor with a carry lookahead adder. The CLA reduced timing for a 16-bit adder as it incurs only a 10 gate delay, compared to the 34 gate delay of the RCA. However, area increases significantly with this more efficient adder.

**5. Exception Handling (0 –2 points):** We implemented NOP / RTI for exception handling. The only exceptions caught by our processor were illegal operations, or siic. This slightly increased overhead. Prior to implementation, illegal operations were just skipped. Now they are handled with a jump to exception handler, and an RTI back to the instruction following the illegal op, with the value 0xBADD loaded into R7.

INST: 0010 0000 0000 0110 j .realstart

INST: 1100 0000 0000 0000 lbi r0, 0

ARCH: REG: 0 VALUE: 0x0000

VERILOG: REG: 0 VALUE: 0x0000

INST: 0001 0000 0000 0000 siic r0

ARCH: REG: 7 VALUE: 0x0010

VERILOG: REG: 7 VALUE: 0x0010

INST: 1100 0111 0001 0000 lbi r7, 16

ARCH: LOAD: ADDR: 0x0010 VALUE: 0xbadd

VERILOG: LOAD: ADDR: 0x0010 VALUE: 0xbadd

INST: 1000 1111 1110 0000 ld r7, r7, 0

ARCH: REG: 7 VALUE: 0xbadd

VERILOG: REG: 7 VALUE: 0xbadd

ARCH: REG: 0 VALUE: 0x000a

VERILOG: REG: 0 VALUE: 0x000a

INST: 0001 1000 0000 0000 rti

ARCH:

VERILOG:

INST: 0100 1000 0000 1010 addi r0, r0, 10

ARCH:

VERILOG:

INST: 0000 0000 0000 0000 halt

SUCCESS: No differences

**7. Synthesis (0-5 points):** For synthesis we were able to synthesize our design without have errors. However, we were not able to meet the timing requirements due to our mem\_system(2 way cache from solution) taking too long. Using the –opt=yes option we were able to get our worst delay to be ~.58 above the limit. We did not know how to go about fixing this. We were using a CLA already. Some ideas we had were to make our forwarding and control hazard logic more optimized, but we did not have the time to do this. Also, we had the though of storing the values from our memory in a register before we used them for combinational logic in order to reduce the path size. This would require us to change how our next pc and logic for passing the instruction to the decode stage would work. Using the solution mem\_system makes it very hard to meet the requirements.