TREMOR

A Triple Modular Redundant Flight Computer and Fault Tolerance Test-bed for WPI's PANSAT Nanosatellite

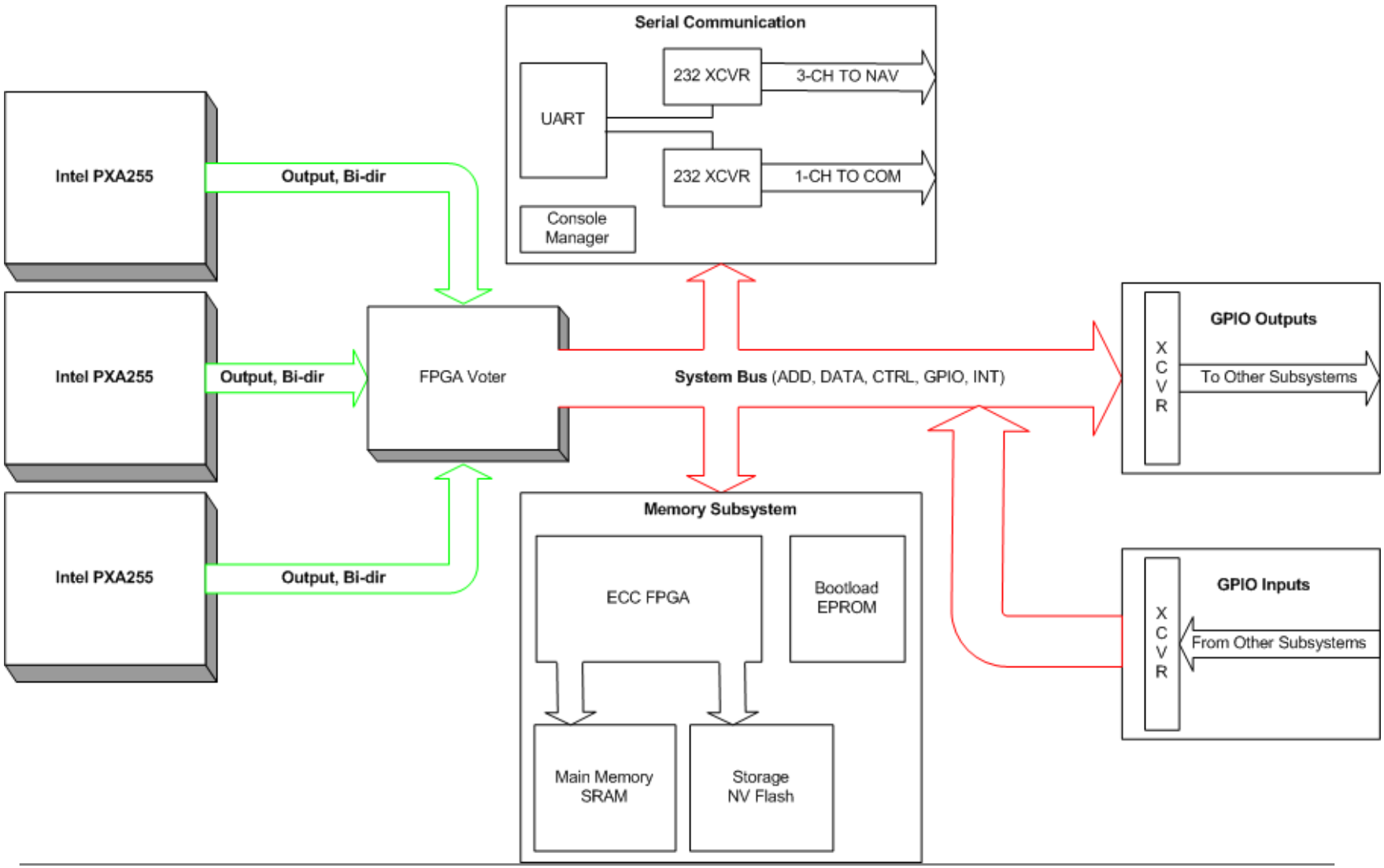
Ryan Angilly
Worcester Polytechnic Institute
ECE B.S. '04
ECE M.S. '06
rangilly@wpi.edu

Advisor: Professor William Michalson
wrm@wpi.edu

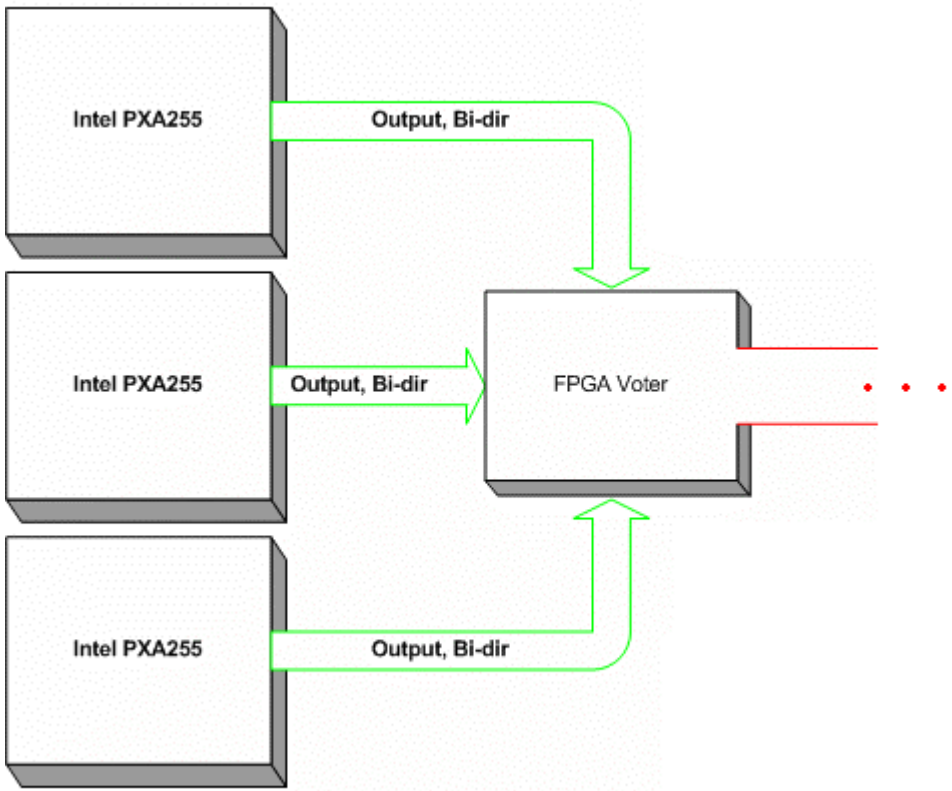
- Reasons for TREMOR
  - Flight Computer for WPI's PANSAT
  - Overcome Single Event Effects (SEE)
    - Single Event Latch-up (SEL)
    - Single Event Upsets (SEU)
  - Use Commercial Off The Shelf (COTS) parts
    - Allows use of modern technology processors
    - Minimizes the need for radiation hardened components

- One copy of memory
- Three processors execute instructions in lockstep – not a distributed architecture
- All processor outputs are fed into a 'Voter' which decides if they agree
  - If in agreement, values are propagated to system bus
  - If not in agreement, majority value is propagated and minority processor is reset and resynchronized at a later time

TREMOR Block Diagram

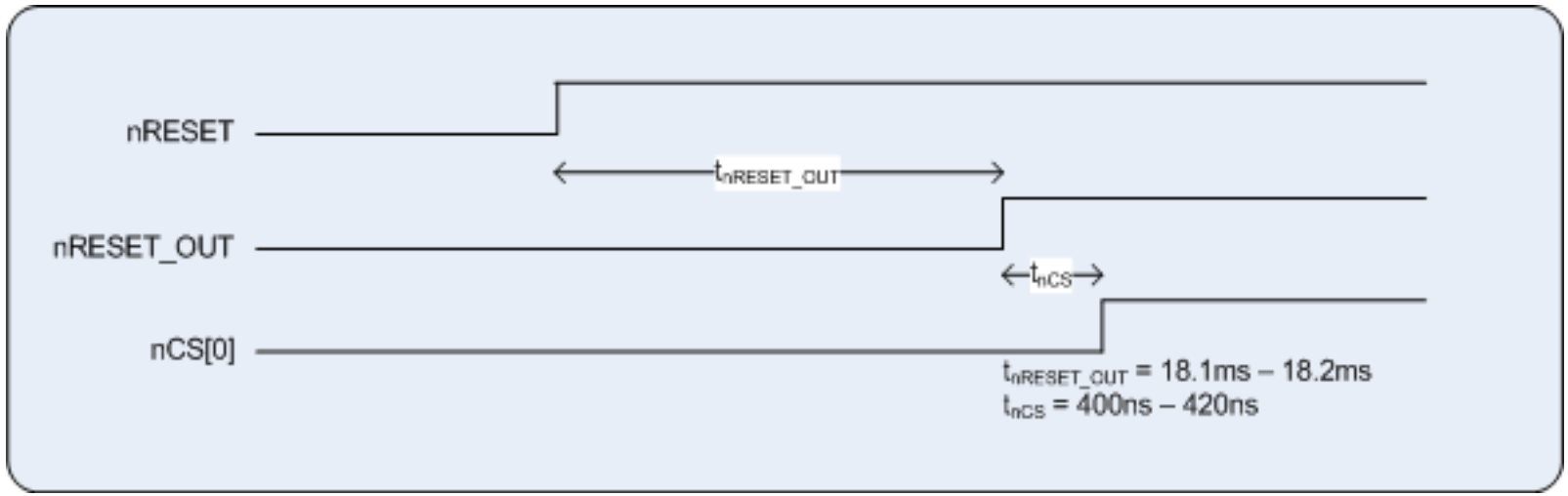


TREMOR Microprocessor Architecture



- Processor Synchronization
  - Timing uncertainty resetting the PXA255 is 100.02us -- unacceptable
- Voting
  - Creating error-free outputs
- Asynchronous Inputs
- Processor Reset
- Processor Resynchronization

Problem: Processor Synchronization



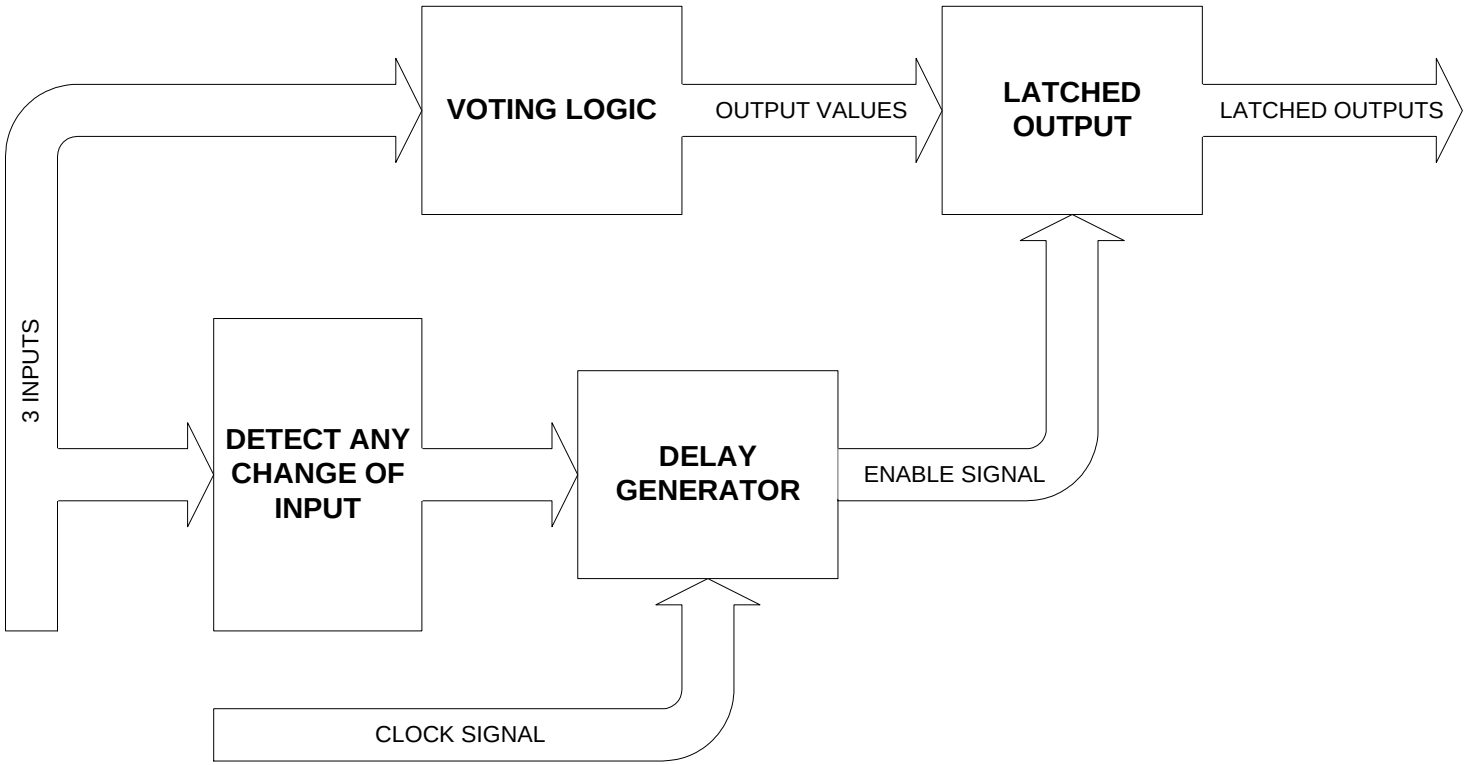
- With an uncertainty of 100.02us and a 199.1 MHz core clock, one processor could be 20,000 clock ticks ahead of another
- However, processors must be synchronized to the point where they can access memory within the same bus cycle
- Question: What is the worst-case synchronization uncertainty between processors?
  - TREMOR Longest Memory Access: 140 ns
  - PXA255 Max Bus Cycle Time: 240 ns
  - Worst case synchronization uncertainty: 100 ns

- The processors will fetch their first instruction at different times
- When a processor fetches its first instruction, setup a state machine in the Voter FPGA to send instructions that change bus cycle time to lowest value (20ns for PXA255)
- Keep sending NOPs to that processor until the other two are in a similar state
- At the same moment, send instructions to all three processors that lengthen the bus cycle time, and jump to reset vector
- Allow FPGA to start running in voting mode

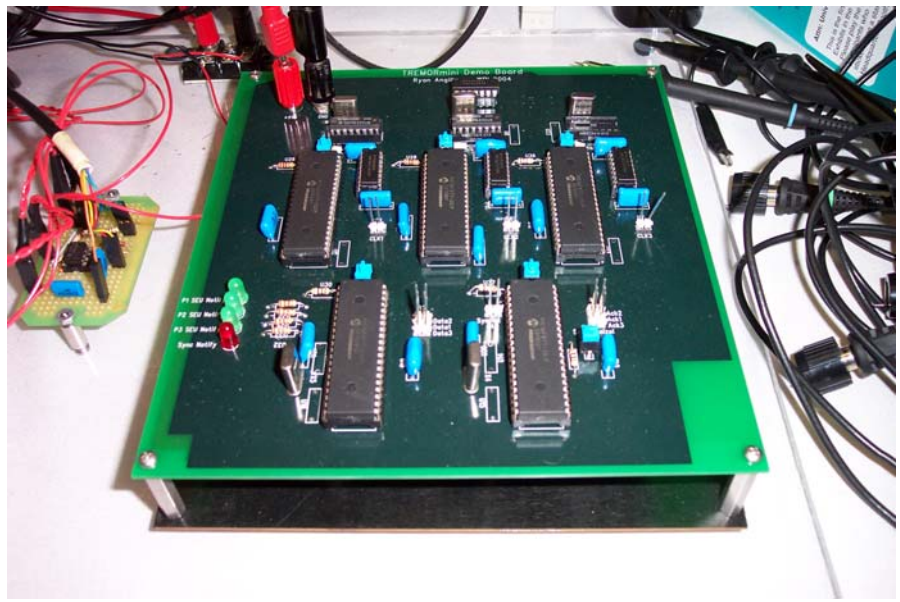
- Uncertainty of 20ns makes it hard to tell the difference between an SEU and a processor that is simply a few nanoseconds ahead of the others

- Solution: Use the 20ns uncertainty to our advantage…

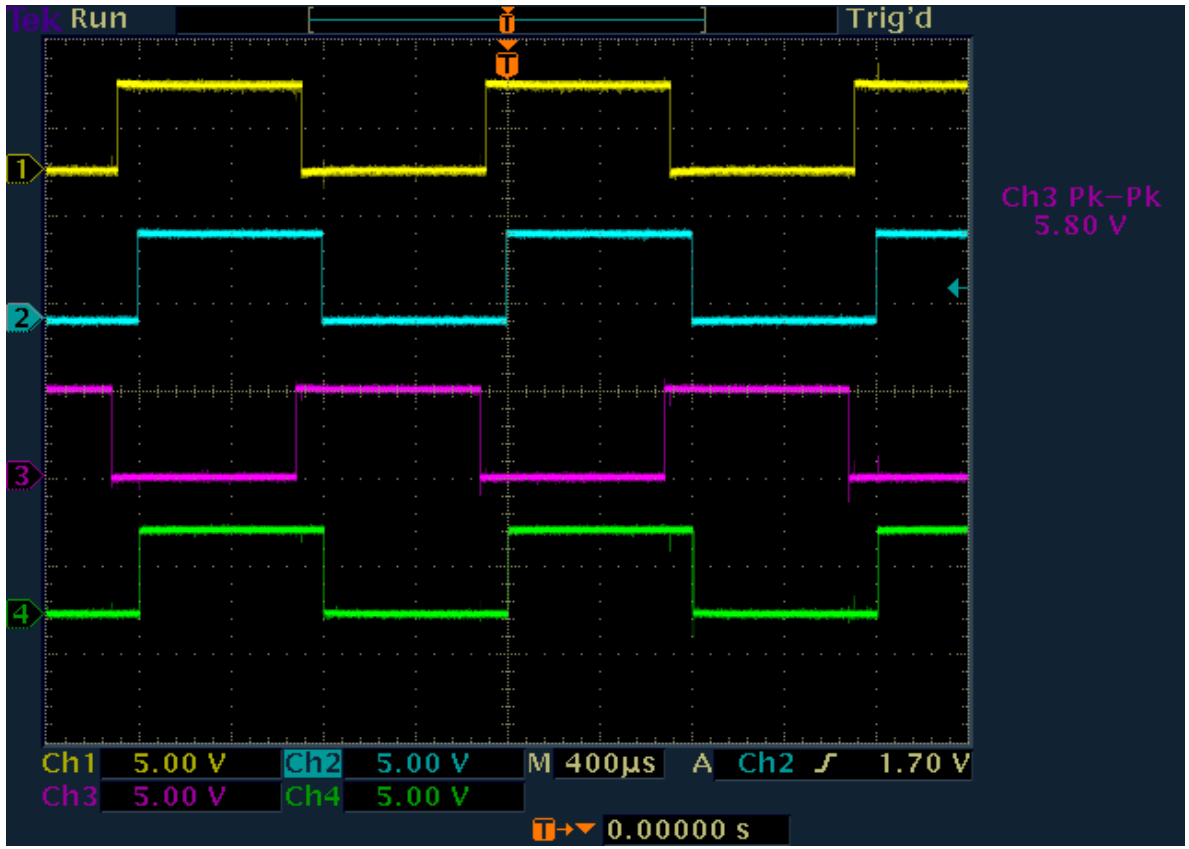
Voter Circuitry



- Proof of concept for two theories behind TREMOR
  - Processor Synchronization Algorithm
  - Blind Voting Algorithm
  - Uses PIC16F877
- 3 Processor PICs
- 1 Synchronization PIC
- 1 Blind Voter PIC

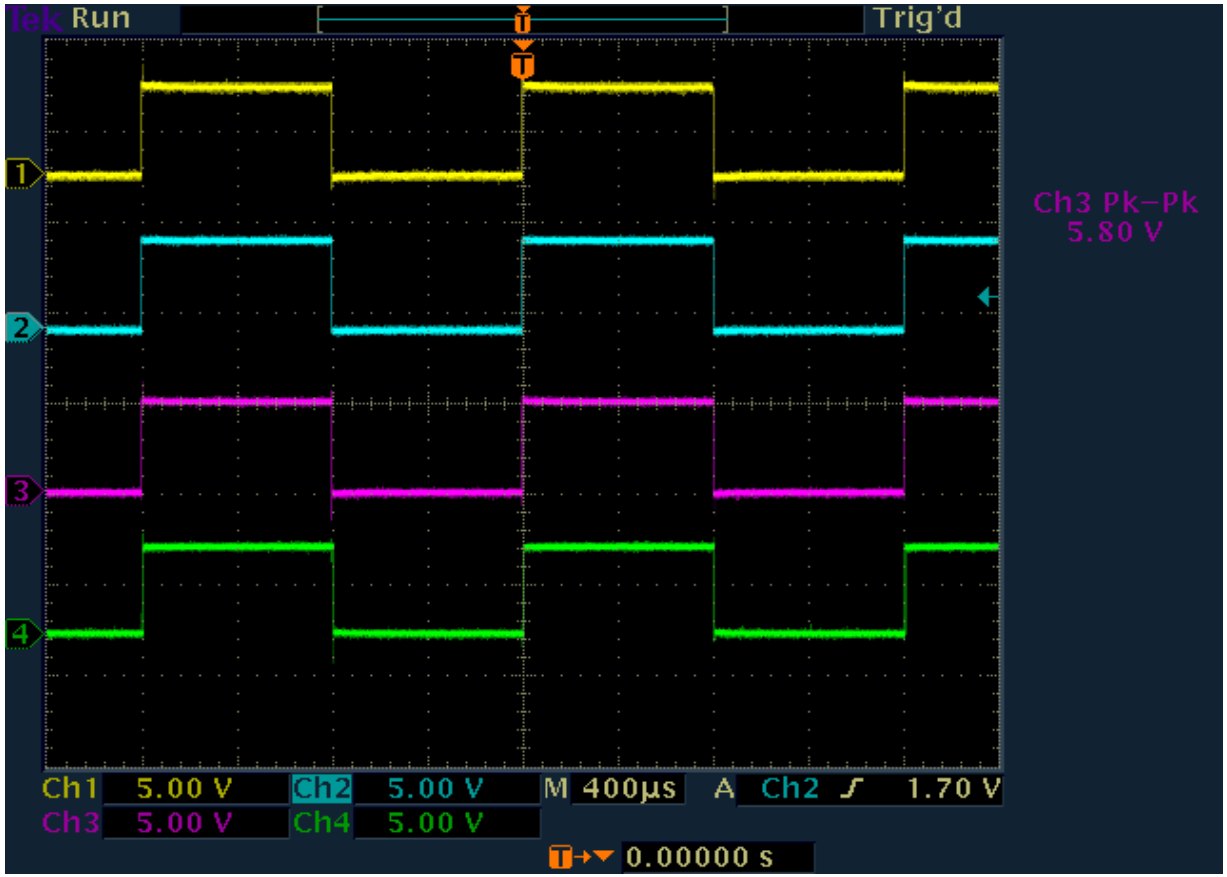


Before Synchronization



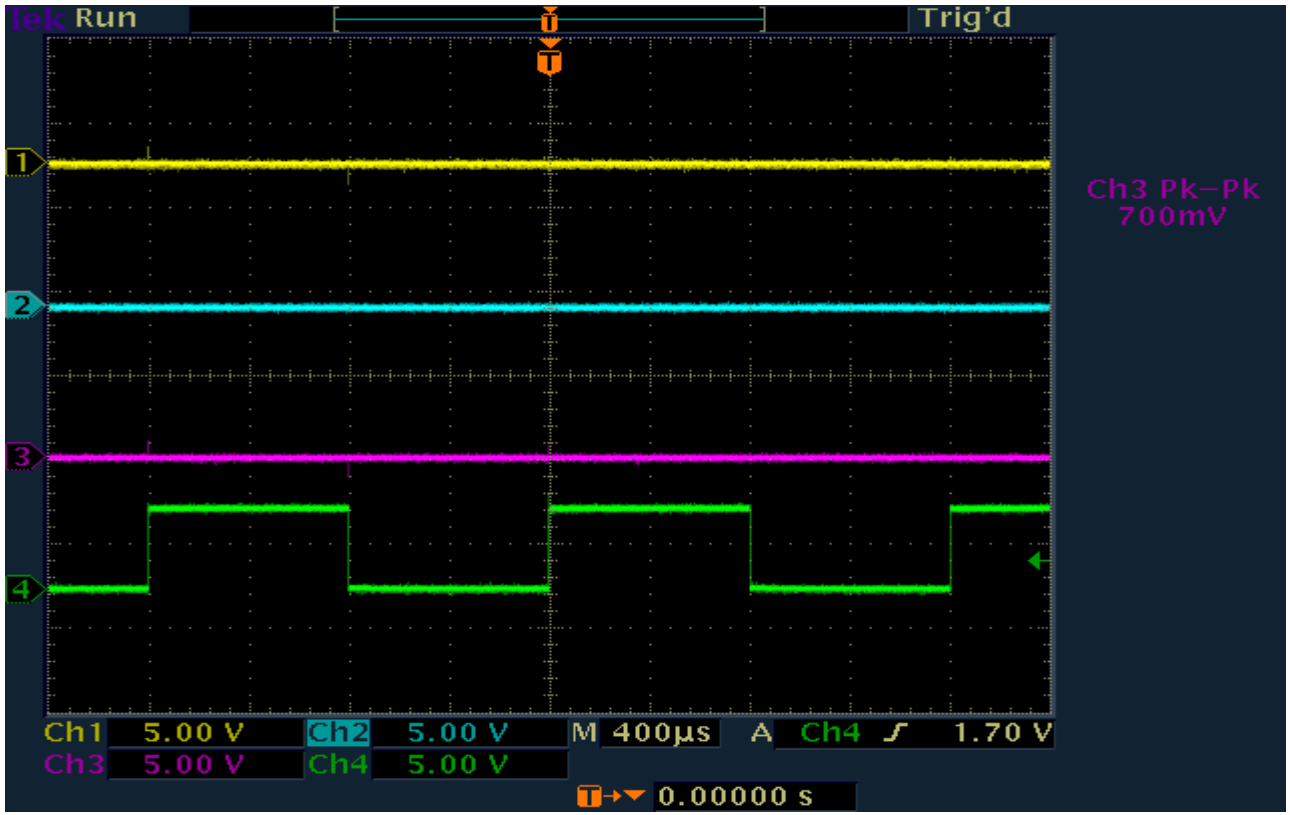
- Before synchronization, the outputs (channels 1,2,3) of processors have random delays. The output (channel 4) is always the majority

After Synchronization



- After synchronization, the processors agree

Synchronous Reset Pin Demonstration



- After synchronization, the processors are never reset (channels 1,2,3) as the majority values change (channel 4)

- Proven that commodity processors can be synchronized for use in TMR architectures
- Demonstrated the ability to vote on processor outputs when clocks are loosely synchronized and unknown to the voter
- Ability to deal with the clock drift associated with cheap crystals/oscillators
- Have started patent application process

I would like to thank
Professor William Michalson
for his advice and guidance

also

Daniel DeBiasio and Mitchell Lauer for their contributions to the project

Questions?