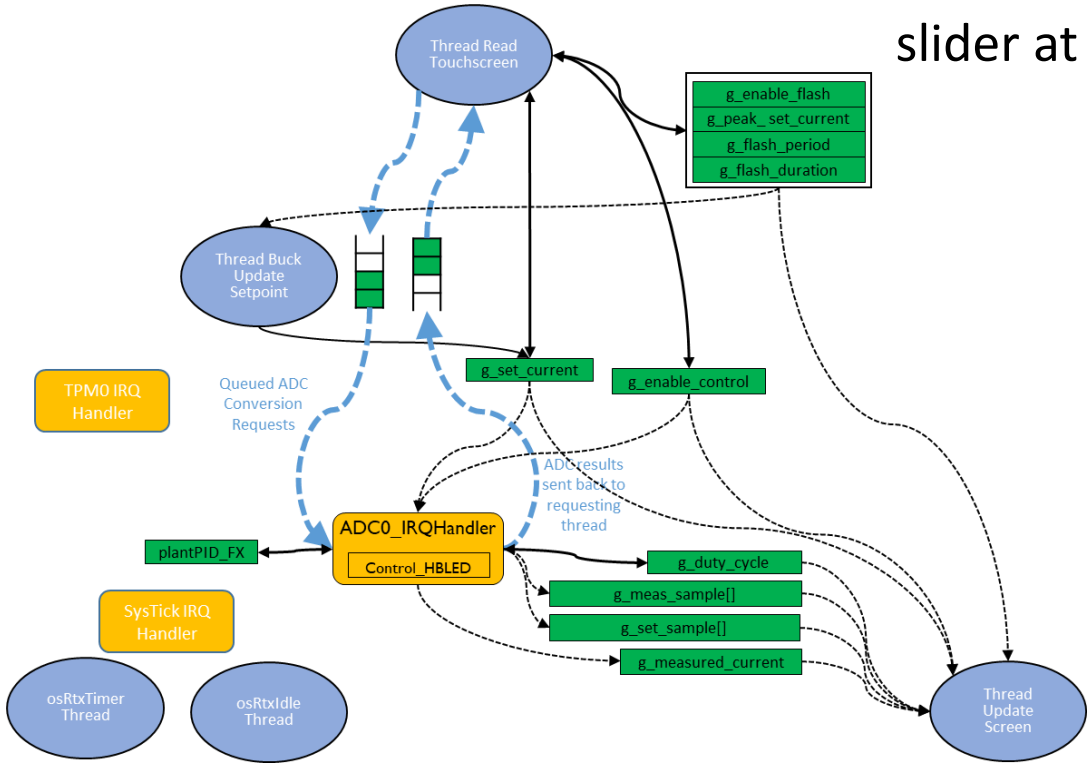
Shield ADC Server

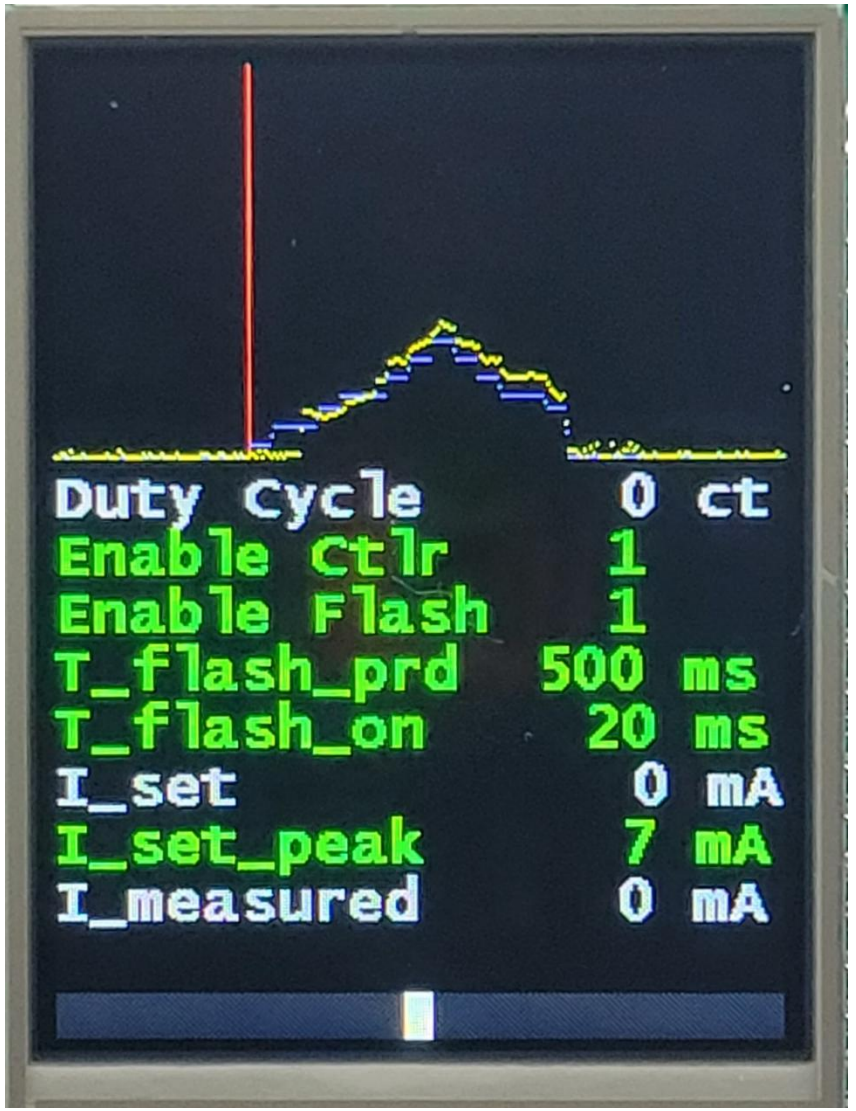
Overview: Two Challenges

- System Overview
- Major Challenges
  - Free up CPU time for rest of application
  - Share ADC between touchscreen driver and buck converter controller

System Overview

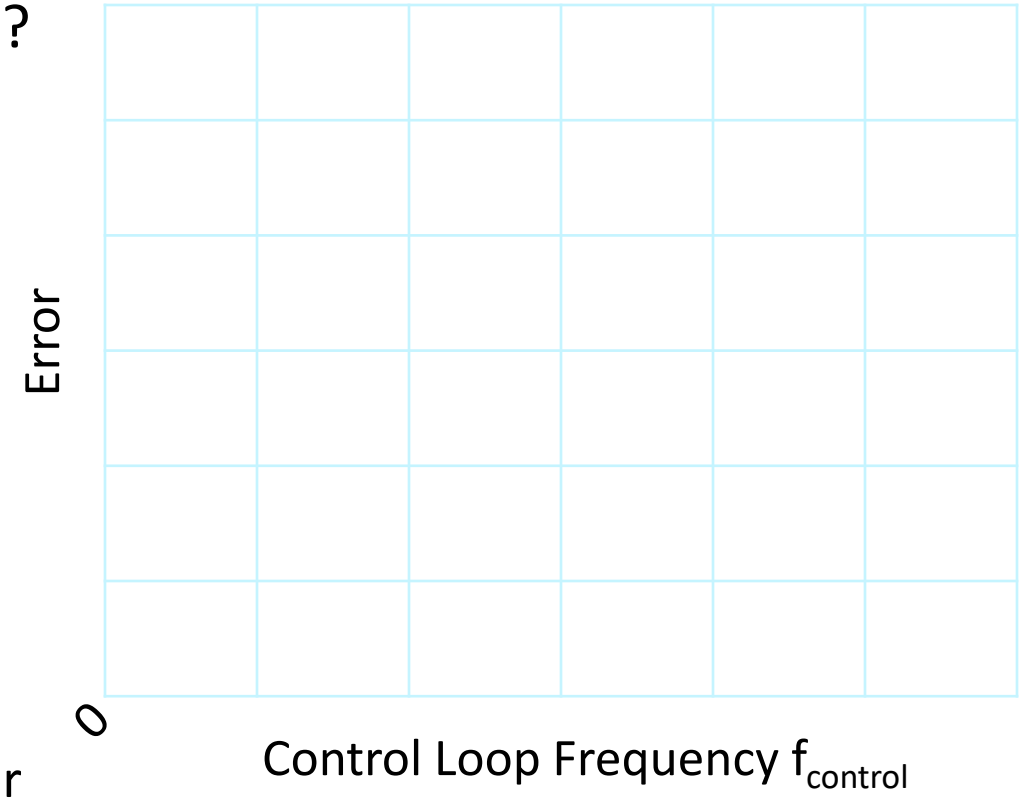
- Bright flashing light!
- LCD Oscilloscope
  - Show LED current setpoint vs. measured value
- User interface (LCD + touchscreen)
  - Monitor and adjust system parameters
  - Select field, then adjust with slider at bottom of LCD



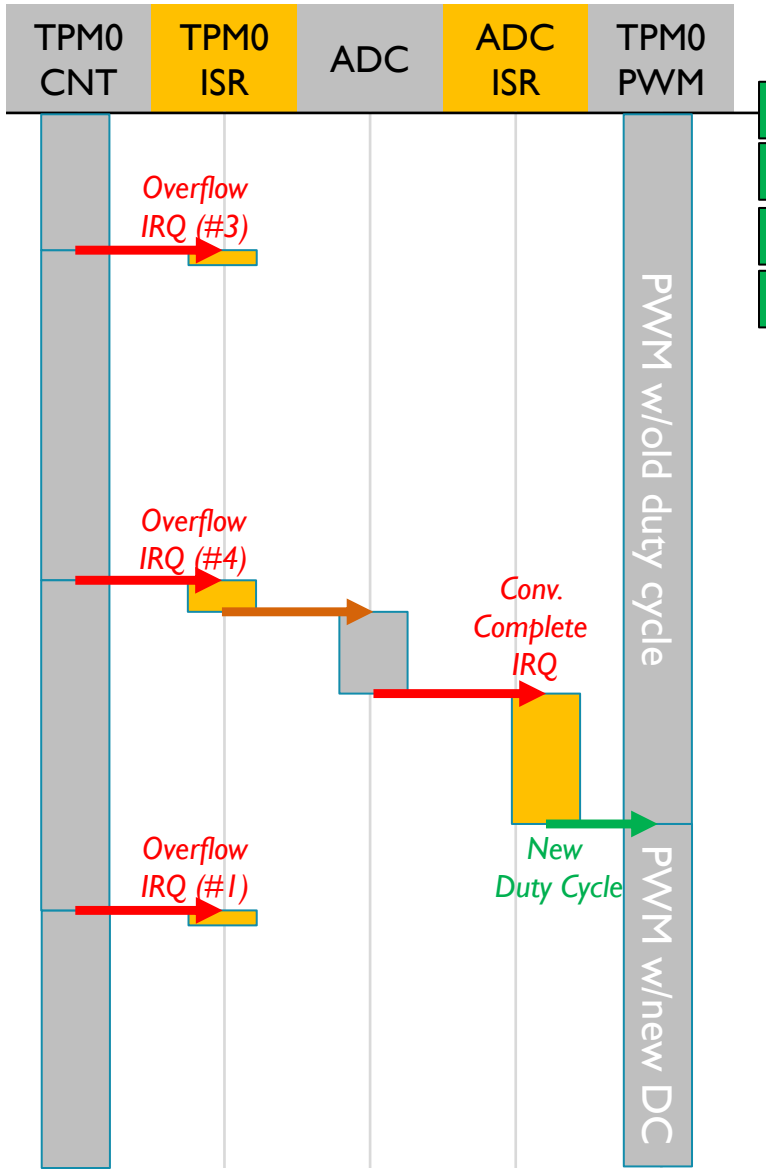


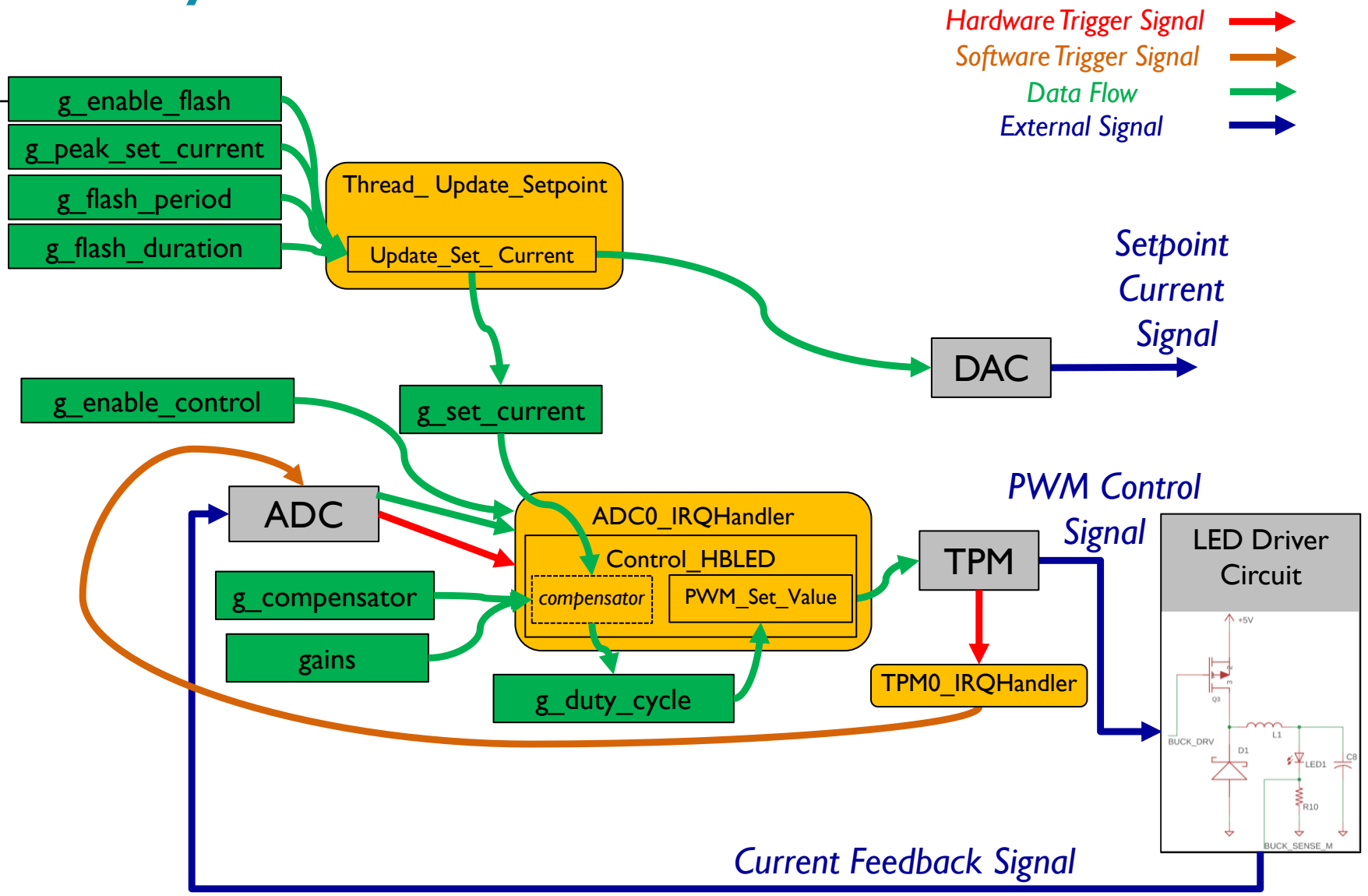
Challenge 1. Need to Free Up CPU Time

- Run control loop on every converter switching cycle?
  - $f_{control} = f_{switching}$, $T_{control} = T_{switching}$
  - Uses most of CPU and ADC time
  - Diminishing returns? How much performance improvement over slower $f_{control}$?
- Run control loop on every Nth converter switching cycle?
  - Keep benefits of high switching frequency: smaller inductors and capacitors
  - Still perform synchronous A/D conversion to reduce error



Trigger ADC on every Nth TPM Overflow with ISR





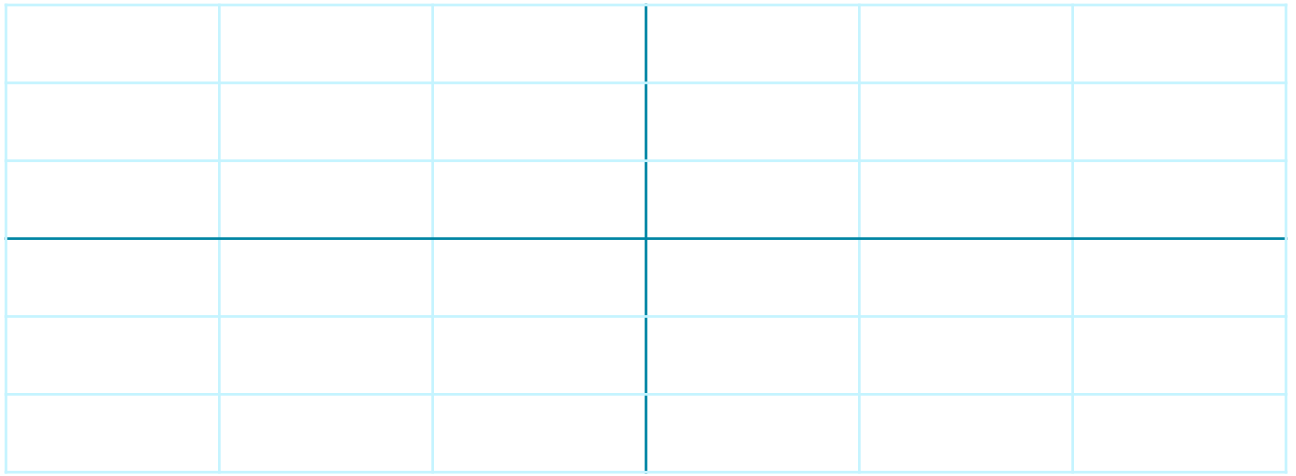
2. Need to Share ADC

- Touchscreen Interface
  - Read X-axis, Y-axis voltages from two resistive voltage dividers
- Constant-Current Controller
  - Read feedback signal indicating current through LED

Examine Timing Characteristics: Touchscreen Interface

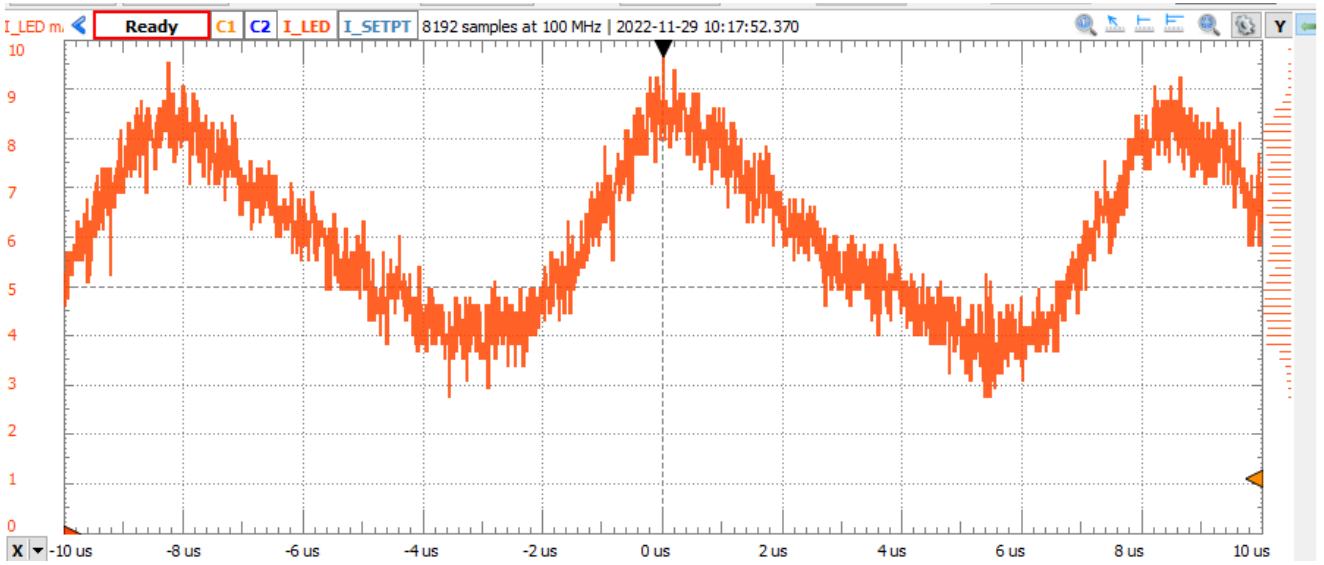
- Signals have no ripple, are essentially DC (flat).
- Even when swiping, signal changes slowly

- Tolerates timing error well
- Delay of up to 100 ms doesn’t affect accuracy significantly

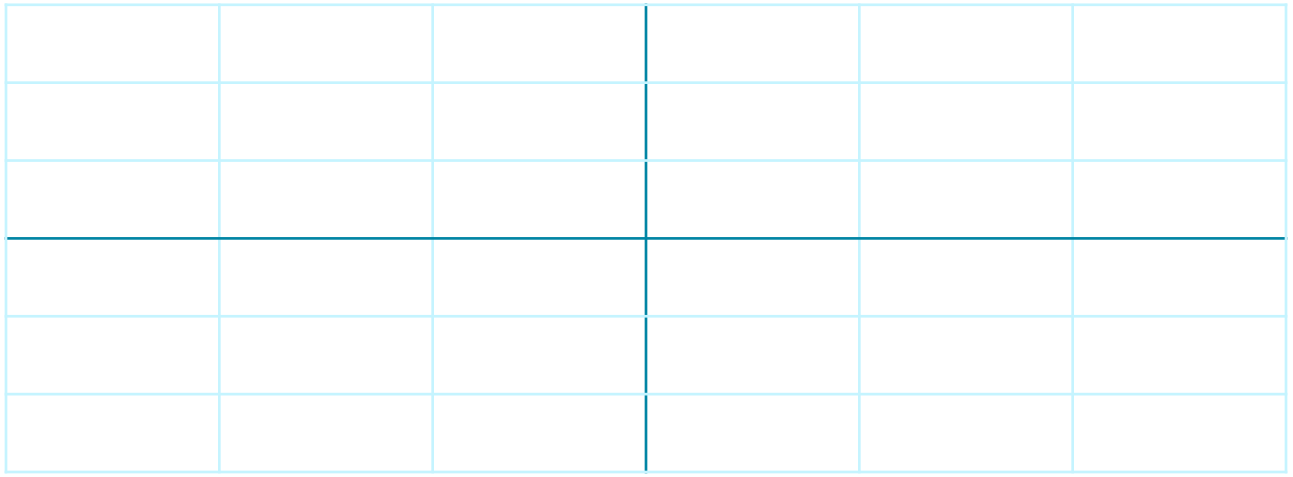


Examine Timing Characteristics: Constant-Current Controller

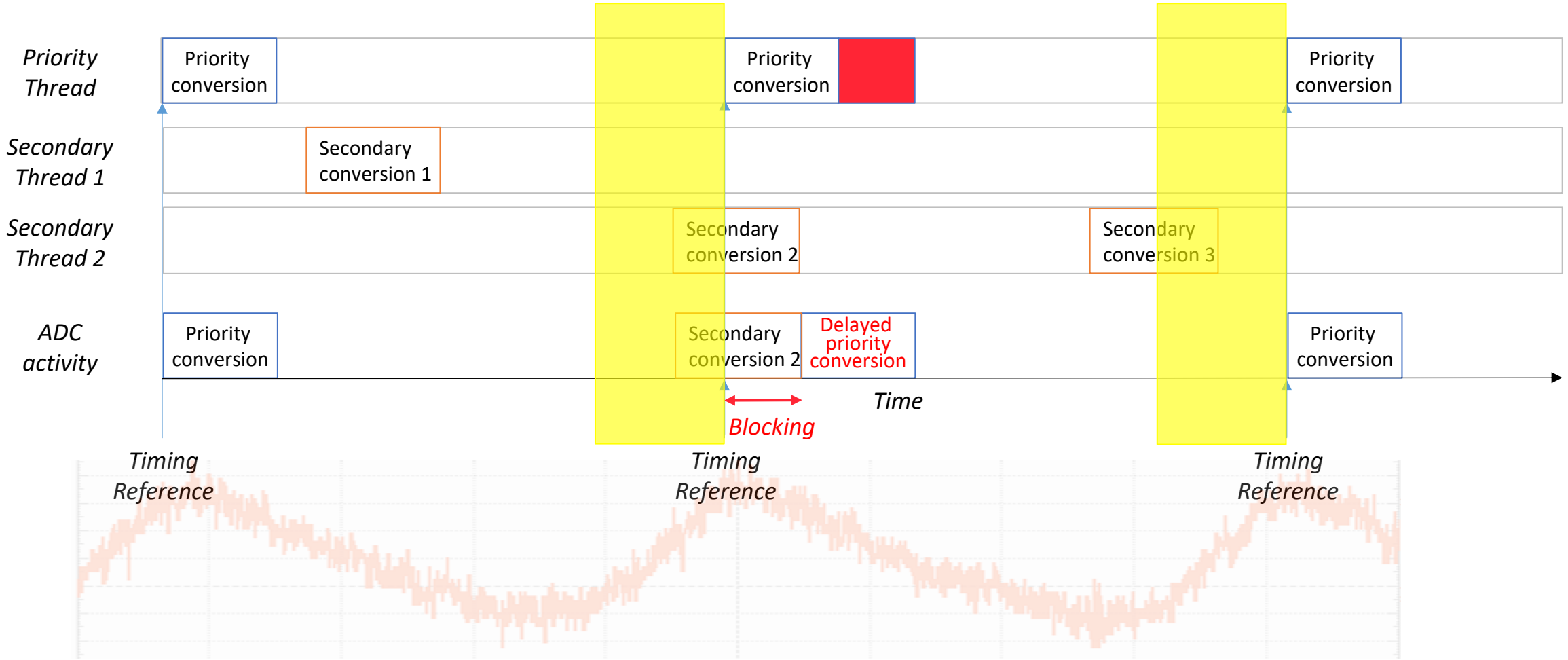
- Signal ripple period is 8 µs for $f_{switching}$=120 kHz
- For synchronous sampling, need very good timing accuracy.



- Doesn't tolerate timing error well
- Error grows, maxes out at ~6 µs (for this waveform)



Share ADC with Mutex?



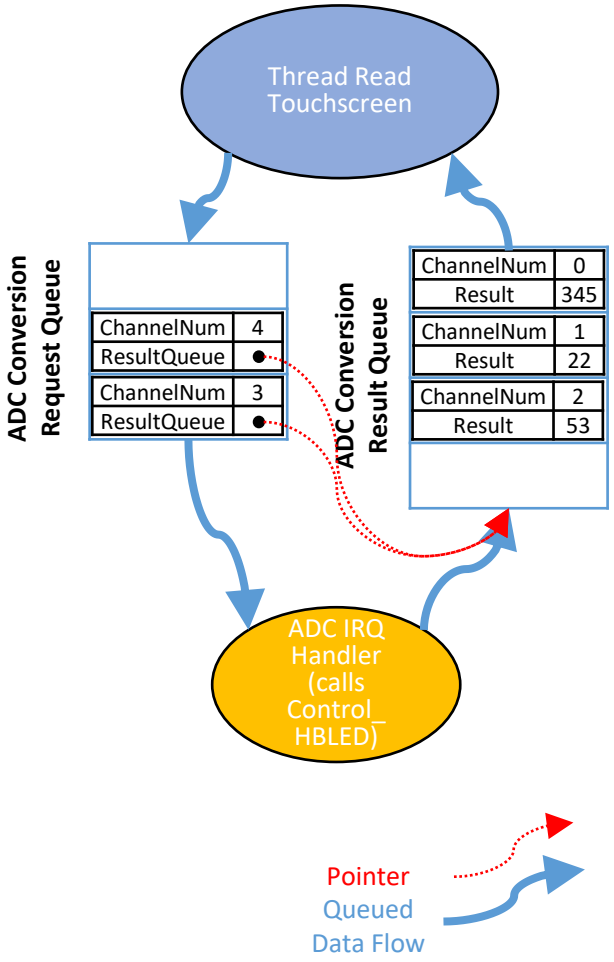
- Secondary conversion starting just before timing reference leads to priority thread blocking, being delayed
  - Timing window of vulnerability

Alternative: ADC Server Thread

- Client threads send non-priority conversion requests to server via request queue
  - “Please convert ADC channel 3, and put the result in this result queue”
- Server sends results back to client threads in specified result queue
  - “Here is the result of your requested conversion”
- Use RTOS message queues to buffer data, trigger threads when ready
- Server thread
  - Blocks on all request queues awaiting requests
- Client threads
  - Enqueue request
  - Block on result queue

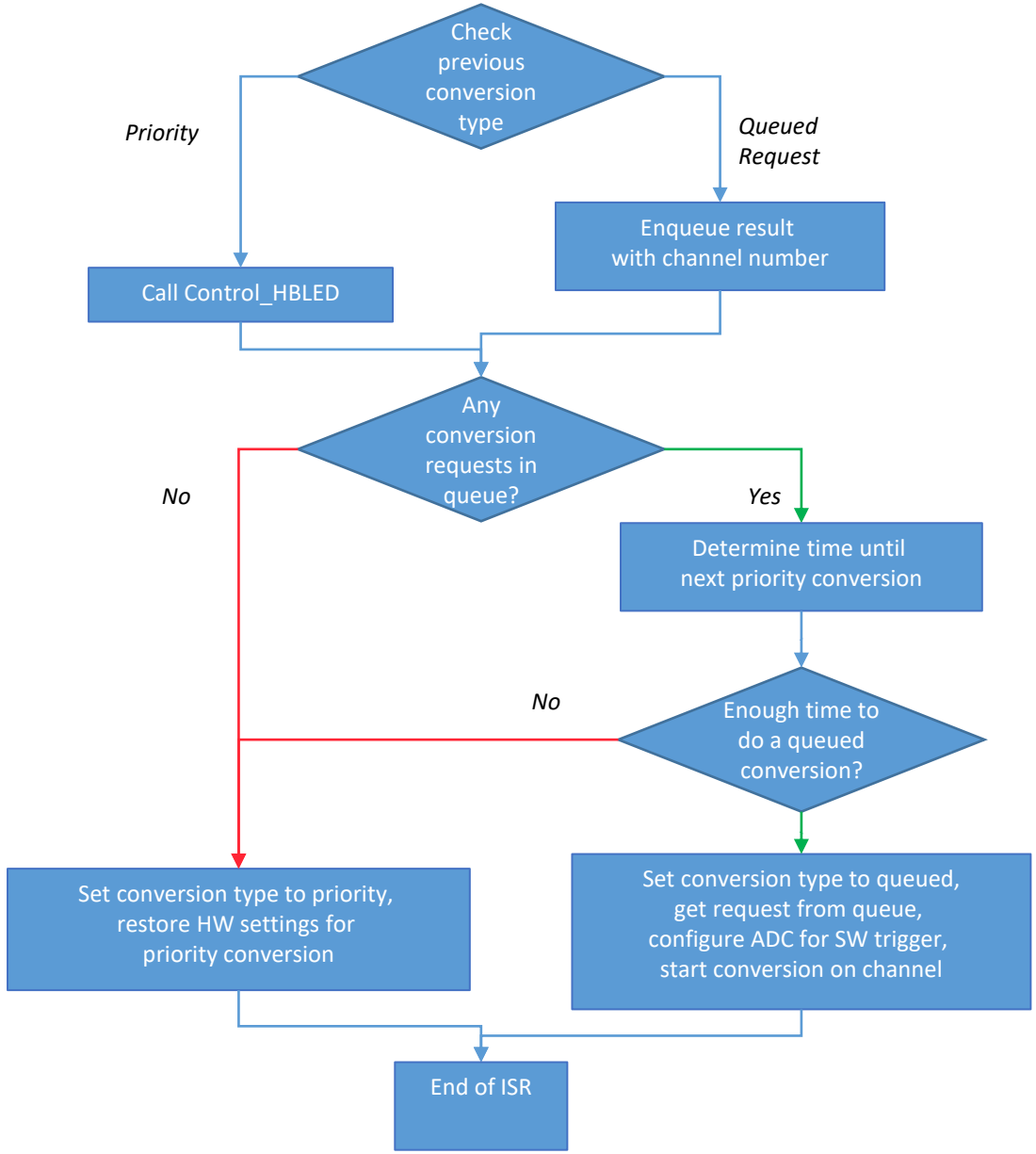
Alternative: ADC Server in ISR

- Server embedded within ADC IRQ handler
  - Runs after each conversion completes
- Priority conversion done directly by server (in this case)
- Secondary threads send non-priority conversion requests to server
  - “Please convert channel 3, and put the result in this message queue”
- Use RTOS message queues to buffer data, signal when ready

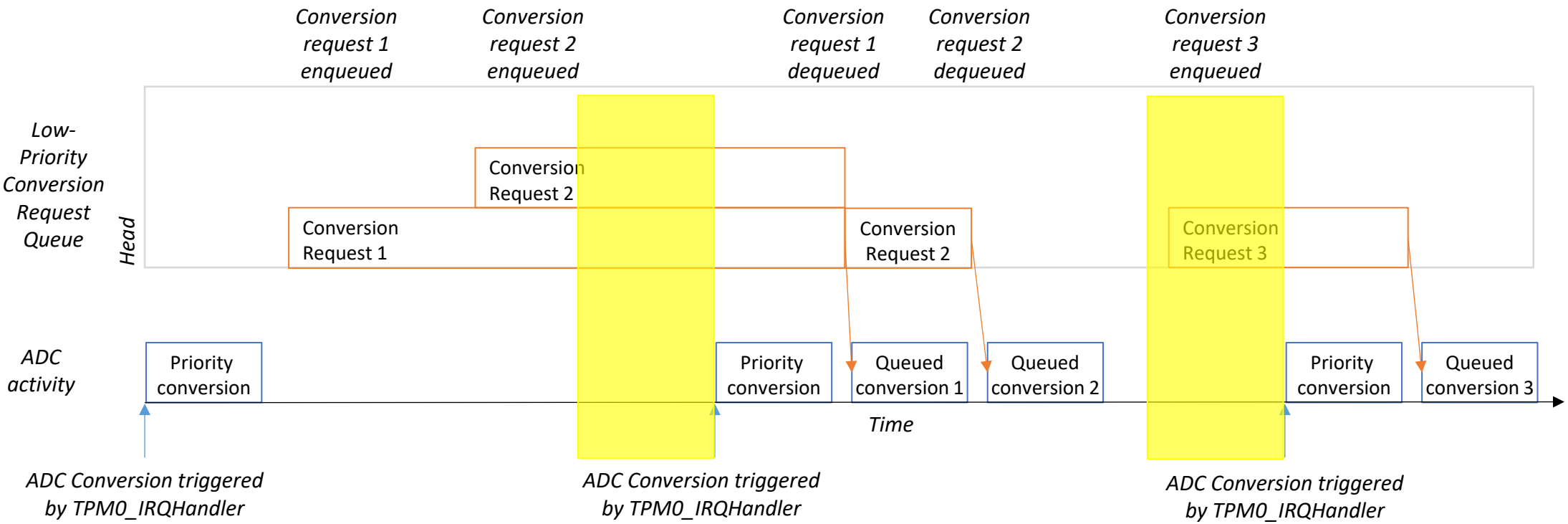


ADC Server Flowchart

- First process result of conversion which just completed
  - Either run control loop or enqueue result
- Then check for pending conversion requests
  - If time available, get conversion request, configure ADC and start conversion
  - Otherwise, configure ADC for priority conversion

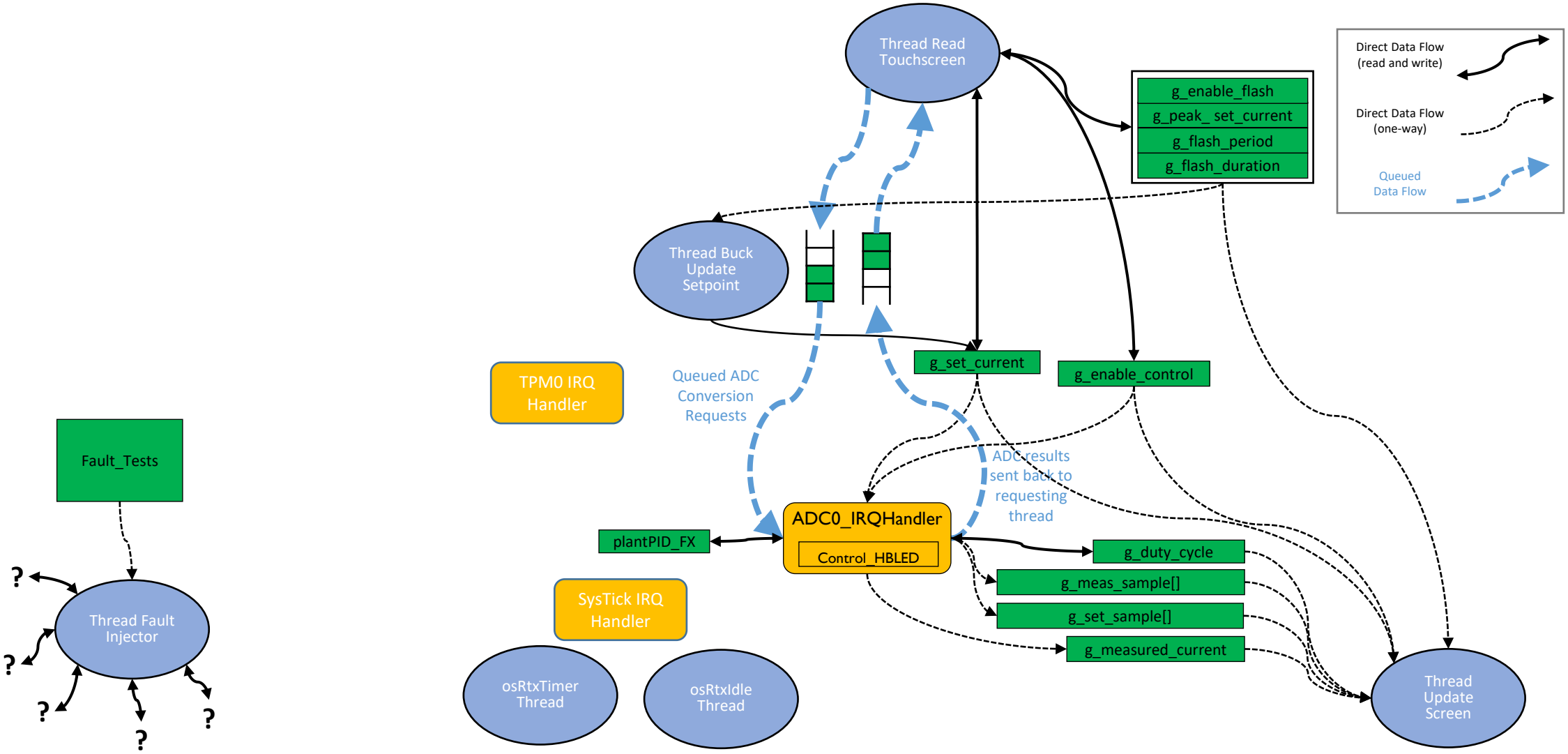


Timing Diagram



Example Code

System Software Overview



Summary

- Free up CPU time
  - Control loop uses most of CPU time, slowing rest of application
  - Little benefit from running control loop as often as converter switching cycle
  - Slow down control loop using timer ISR to start ADC on every Nth timer overflow
    - Other approaches possible based on available hardware: chain hardware timers, run two timers in parallel, DMA control…
- Share ADC between time-sensitive users
  - For synchronous sampling, must limit timing error of ADC sampling
  - ADC conversion for a secondary channel would delay priority sampling too much
  - Use server software
    - Accept conversion requests
    - Service priority requests on-time, others when time is available)
  - Use RTOS message queue to support multiple pending conversion requests and responses
  - Design Optimization: Merge server into ADC ISR, as it will be running at least once every control loop ($T_{control}$)