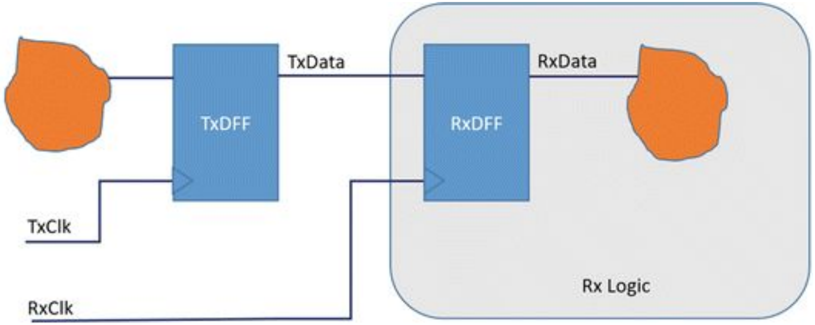
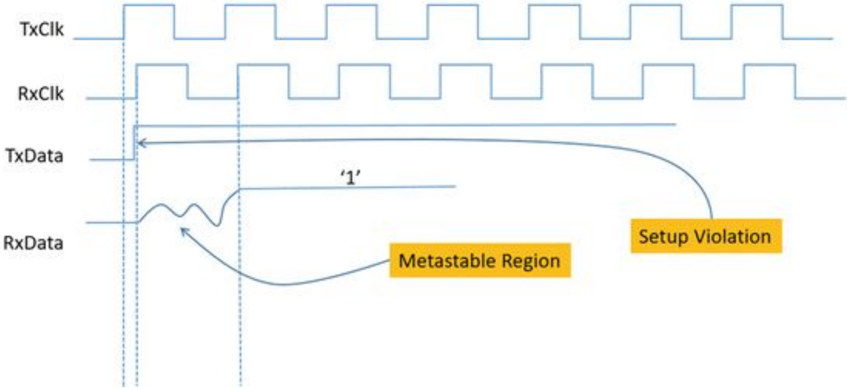
Handshake Protocol and Channel Bonding in Yarr firmware

Lauren Choquer - University of Washington ACME Lab / LBNL
Timon Heim - LBNL

Metastability

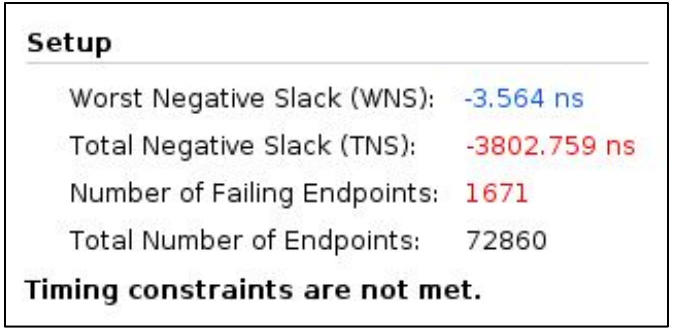
- Asynchronous signal being passed between clock domains of different frequencies
  - System can be sent into an unknown state for an unbounded amount of time (halfway between 0 or 1)
- Problem if setup (before clock edge) or hold (after clock edge) periods violated
- One metastable signal can affect entire system





Metastability in Yarr-fw

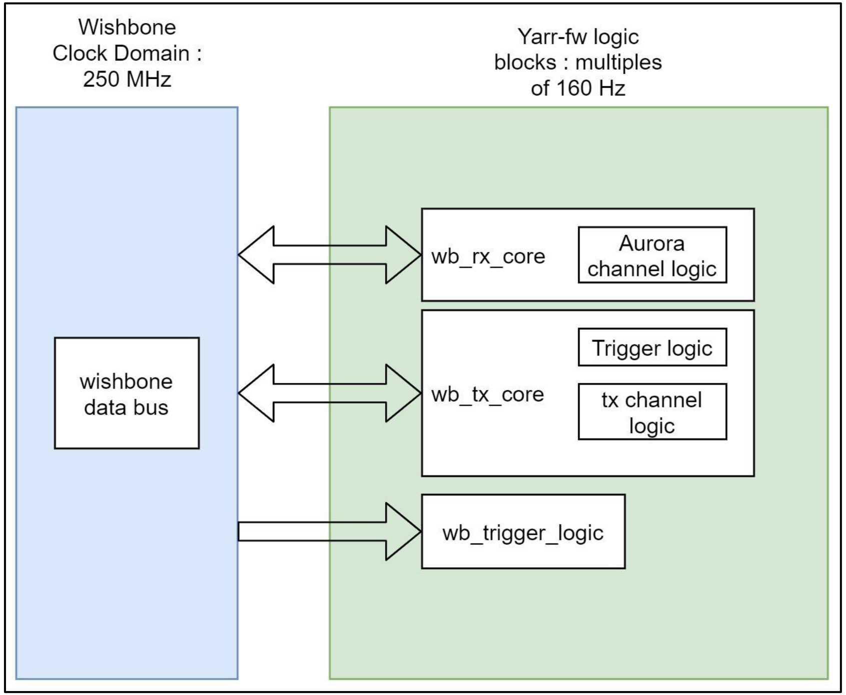
- Conflicting clock domains with status registers passing in between
- Metastable signals changing rarely
  - In theory, not much of a problem for our system
  - Still must be resolved to pass timing analysis at proper clock speeds



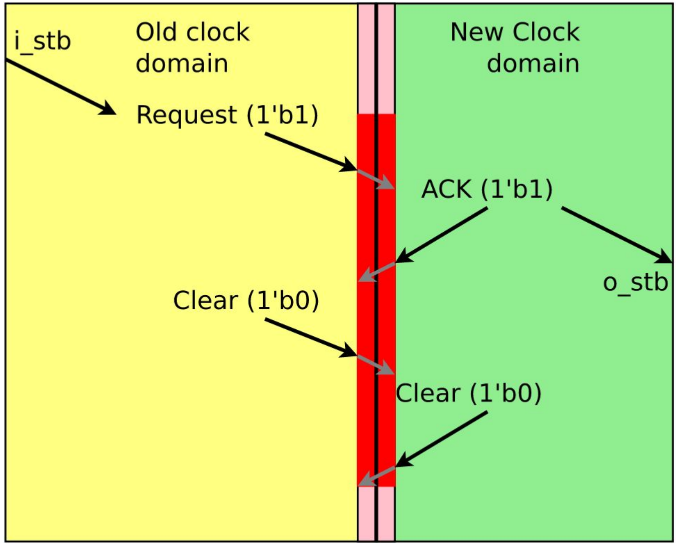
```
Setup
    Worst Negative Slack (WNS):    -3.564 ns
    Total Negative Slack (TNS):    -3802.759 ns
    Number of Failing Endpoints:   1671
    Total Number of Endpoints:     72860
Timing constraints are not met.
```

Yarr-fw Wishbone Express

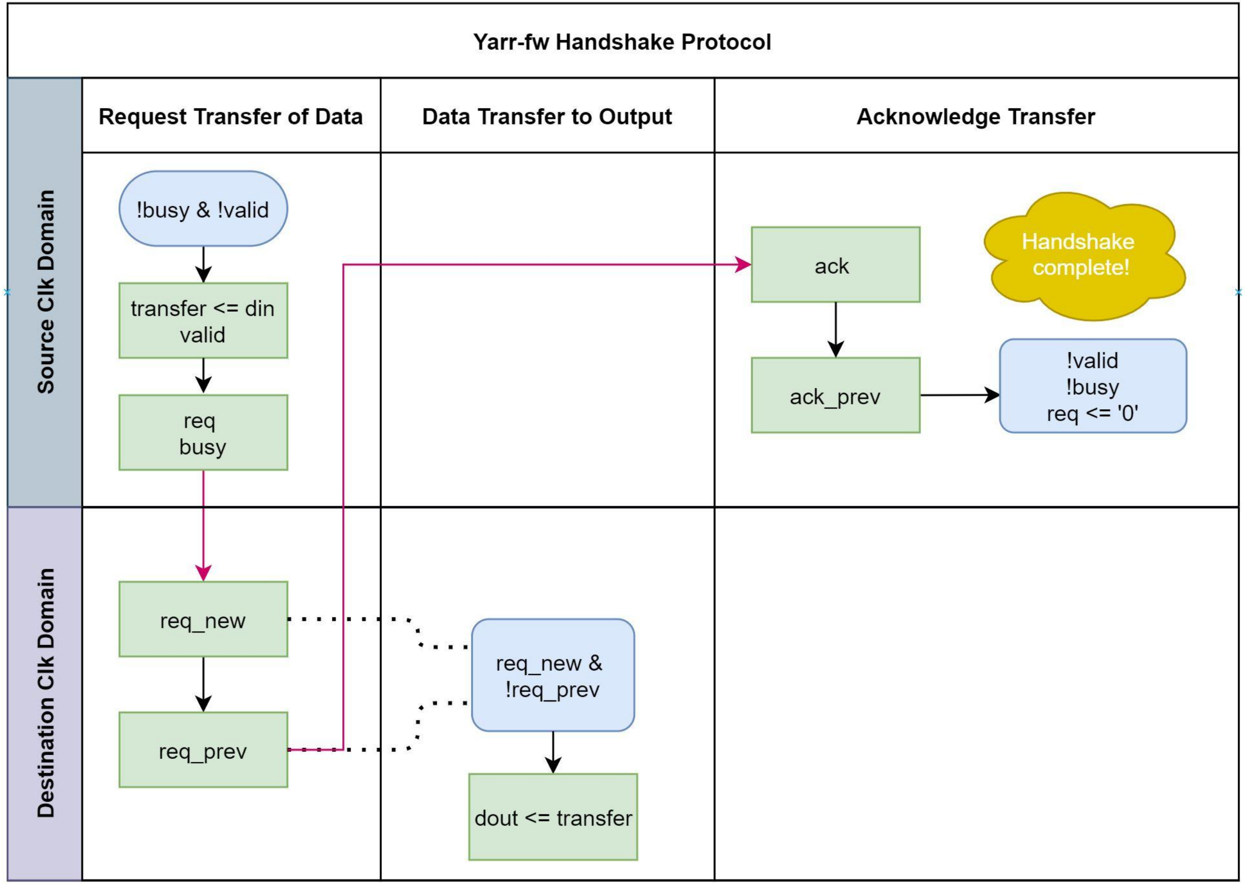
- High-speed, flexible interconnect
  - AXI-stream -> wishbone -> firmware logic blocks
- Unrelated clock frequencies



Handshake Protocol



- Ideal synchronization method for crossing between clocks of non-integer ratios
- Preferable for multi-bit signals compared to flip-flop chain synchronizer
- Lower resource utilization than FIFOs, preferable in Yarr-fw case



Key differences:

- Addition of transfer data register for multi-bit signals
- Pre-transfer valid signal
- Raise 'acknowledge' flag once data transfer is complete

Current Status of Handshake

- Implemented in Yarr-fw
    - Code has been reviewed, will be merged soon
- Passes timing constraints in Xilinx Vivado
- Has been stress-tested on kc705 FPGA
- Next step: scanConsole verification using RD53A

KC705 Yarr-fw configuration

- Yarr-fw previously set up only for **xpressk7** and **tef1001** boards
- Added support for kc705 hardware
  - New set of constraint files
  - Flash script updated
  - Currently debugging with RD53A

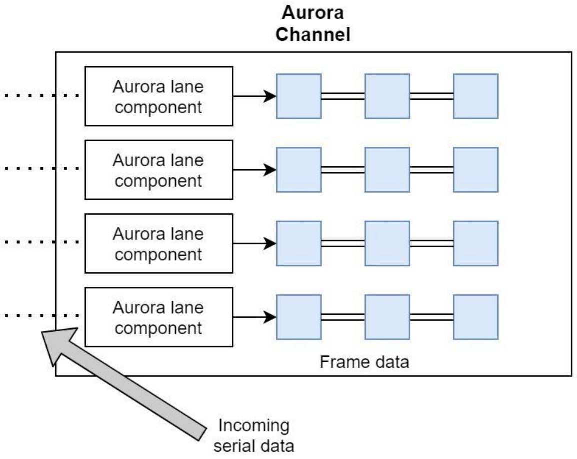
CERN SPEC (Spartan 6)

ReflexCES XpressK7

Xilinx KC705

Trenz TEF-1001

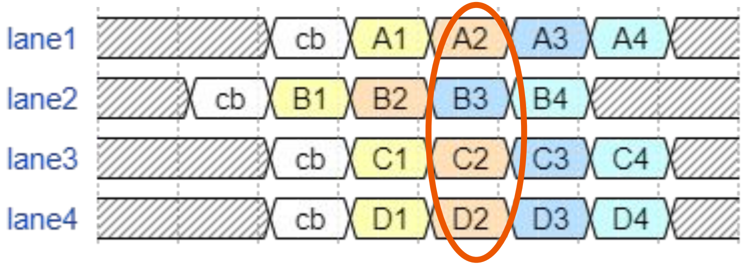
Aurora in Yarr-fw



- Aurora interface: “lightweight link-layer protocol that can be used to move data point-to-point across one or more high-speed serial lanes”
  - Using 64/66b encoding
- Yarr-fw uses up to 4 Aurora channels
  - Each consisting of up to 4 lanes
- Each lane produces 64b-frames of data/commands with 2b headers

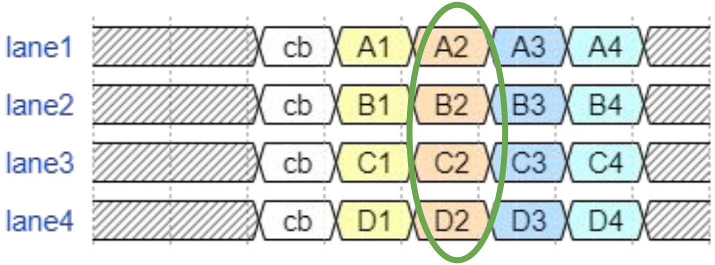
Channel Bonding

Lanes can become out of sync:



Misaligned data

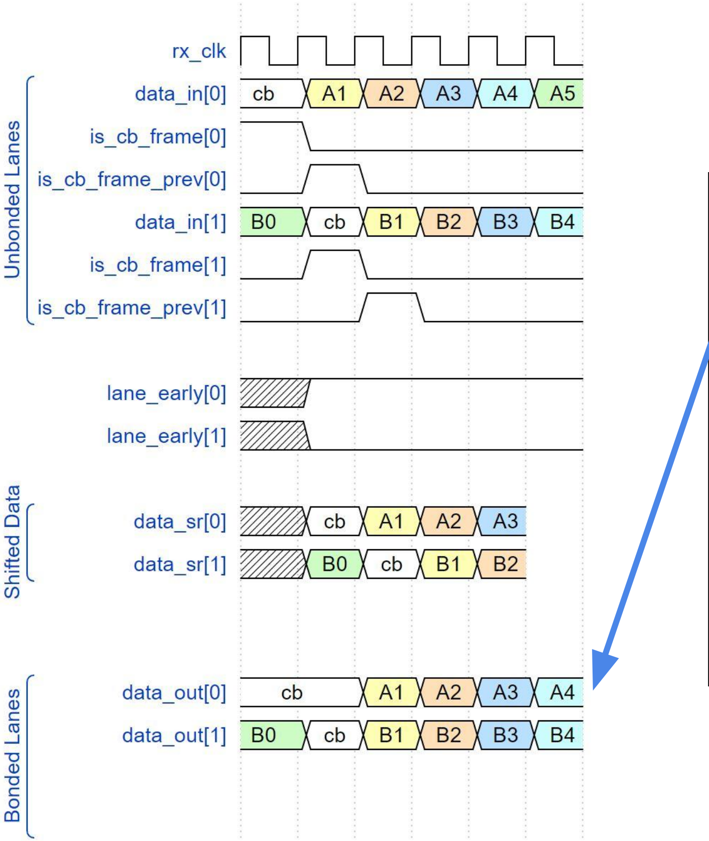
For RD53B, a **channel bonding** process is needed in order to realign them:

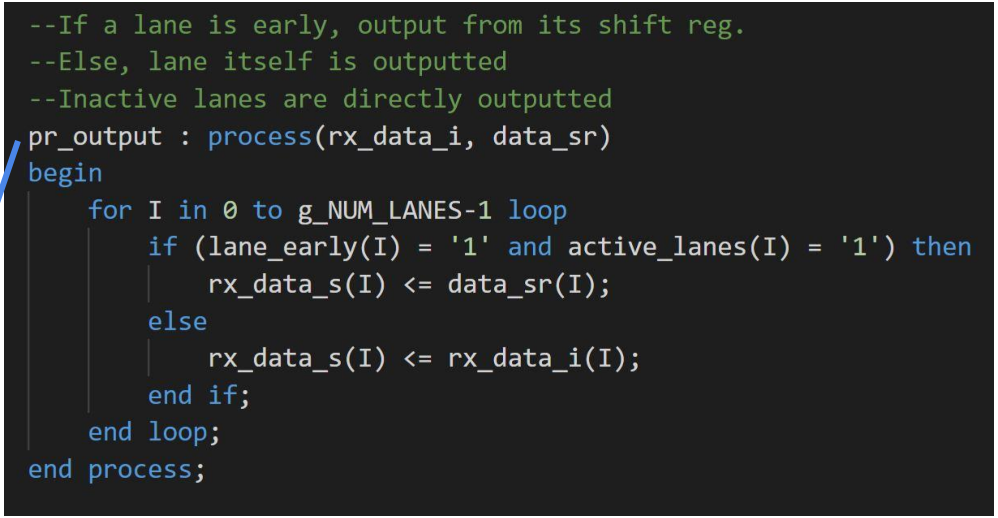


Correct data alignment

Channel Bonding Implementation

- Channel bonding command frames:
  - command header = 0b10
  - idle tag = 0x78
  - channel bonding flag raised high
- Method:
  - Detect channel bonding frames
  - Based on CB frames, detect which lanes are early compared to rest
  - Delay early lanes using shift register





```
--If a lane is early, output from its shift reg.
--Else, lane itself is outputted
--Inactive lanes are directly outputted
pr_output : process(rx_data_i, data_sr)
begin
    for I in 0 to g_NUM_LANES-1 loop
        if (lane_early(I) = '1' and active_lanes(I) = '1') then
            rx_data_s(I) <= data_sr(I);
        else
            rx_data_s(I) <= rx_data_i(I);
        end if;
    end loop;
end process;
```

Channel bonding simulation waveform and output logic

Thanks for listening!

Questions?