Deep Sleep Idle Concept for PHYs

George Zimmerman

Solarflare Communications

Introduction

- Deep Sleep Idle Overview
- Transition times
- Deep Sleep Idle concept
- 10GBASE-T Deep Sleep Idle
- Power& Startup time estimates

Deep Sleep Idle Overview

- Return to old-fashioned ethernet
  - Transmit when data is there to send
  - Quiescient when it’s not
- Maximize energy efficiency
  - Energy per bit will be lowest at the highest speed
    - May not be true comparing 1st generation of the new speed, but becomes true rapidly
  - Also true for optical
- Problem: how to transition rapidly

Transition Times

- EEE Study group focused on 1 msec transition times
  - Improving startup and bypassing autoneg would be sufficient for this timescale
  - Buffer sizes and application latencies will suffer
- Chipset and Computer architecture power management focus on microsecond times
  - Ultimately, PHY latency and propagation time limit transition times

Transition Times (2)

- Deep Sleep Idle transition time will be limited by startup/”sync” time
  - Best techniques will maintain PCS and PMA synchronization
    - Enable blind return to high rate
    - Limited only by block sizes, latency and prop delay
  - Each PHY rate (100BASE-TX, 1000BASE-T, 10GBASE-T) uses its own PCS & PMA

    -> each has its own best deep sleep idle state!

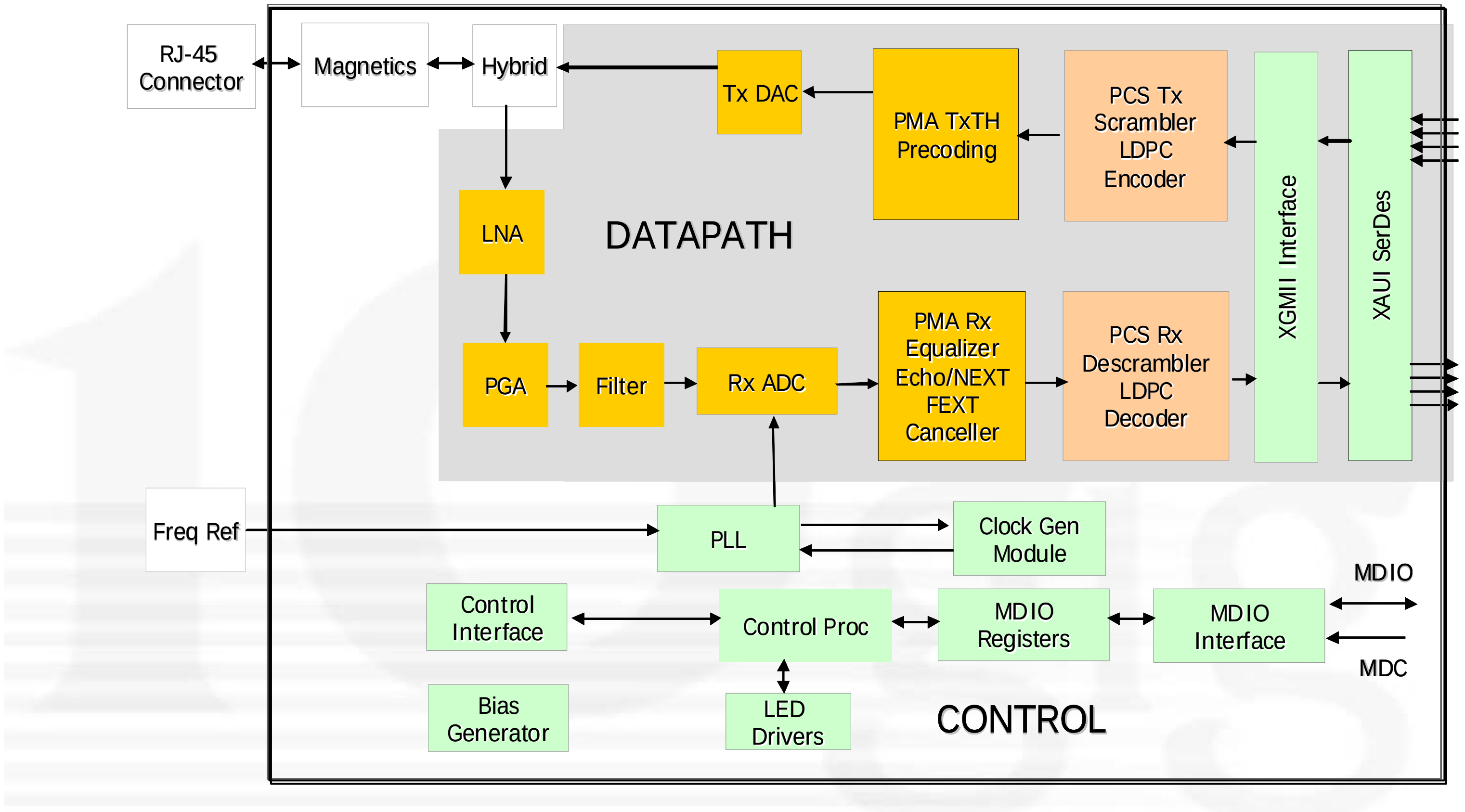
Deep Sleep Idle Concept

- Add a counting state machine for deep sleep idle modes to wake up periodically
  - Turn off receivers, transmitters for N frames
  - Turn on receiver (or transmitter) on schedule for 1 (or M) frames
    - Check for “wake-up” codeword
    - Continue activity transitioning back to active mode or go back to “counting sleep” depending on codeword received
- Minimizes Standards work, minimizes power, minimizes return time
  - Maintains PMA & PCS structures
    - Vendor-dependent hardware scheduling
  - Standards Work is focused on negotiation, state transitions and parameters of the deep sleep idle state

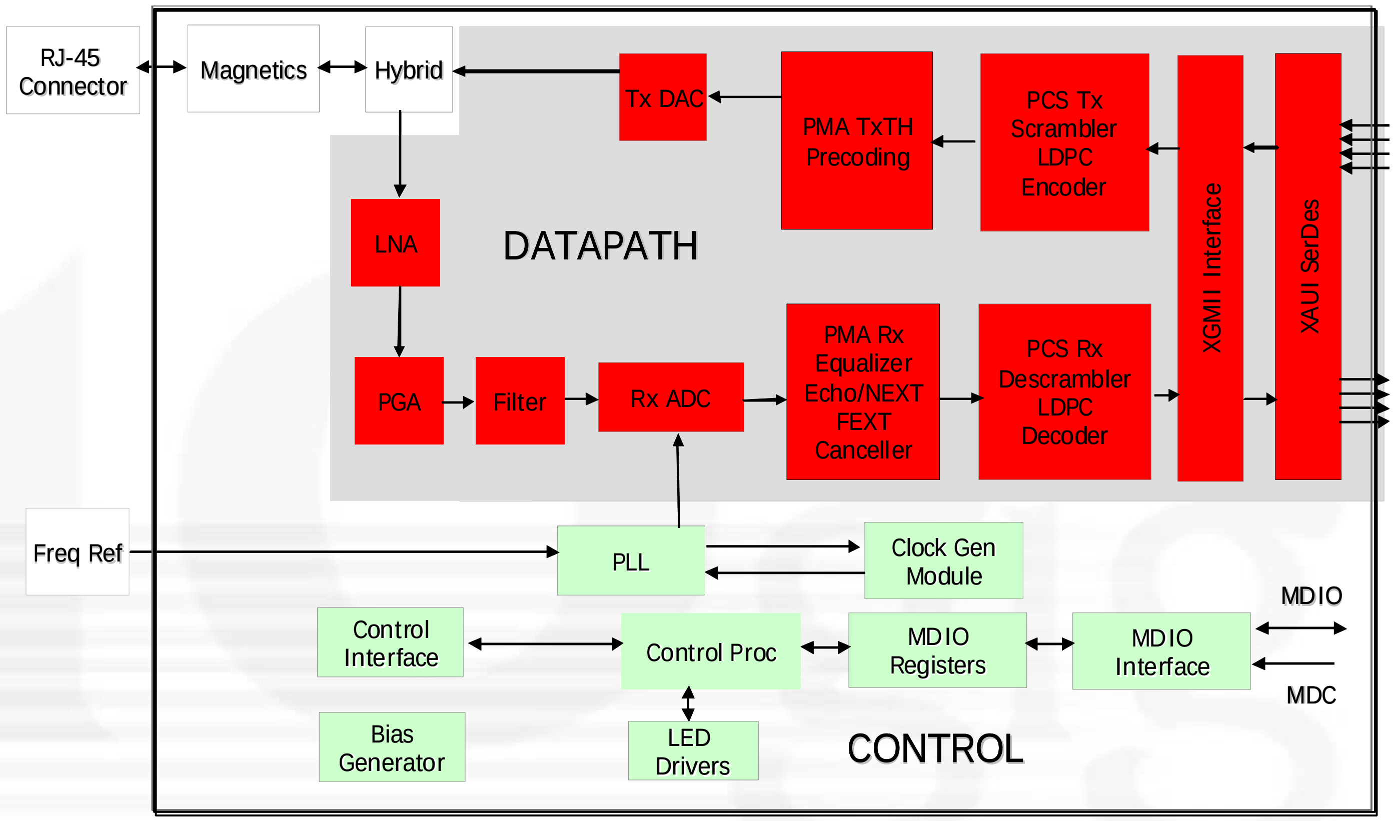
10GBASE-T Deep Sleep Idle

- 10GBASE-T utilizes a fixed-length LDPC PHY frame
  - Set framing interval makes it easy to signal a reduced frame speed
    - Receiver can flywheel between “off” frames without burning signal processing or decoding power
    - Near and far-end frames can be staggered for additional savings when frames are received
  - Return to high speed can be signalled and continued within a single frame
- Other rate PHYs can be structured similarly using periodic IDLE symbols and their PCSs

10GBASE-T Transceiver - On state



10GBASE-T Transceiver – Deep Sleep Idle state



RED BLOCKS ARE POWERED DOWN OR HAVE CLOCKS GATED BETWEEN PULSED FRAMES

Power Estimates

- Off_frame power: P_off = P_clkgen+P_ctl+P_ovr
  - P_ovr: vendor-specific overhead for leakage & housekeeping (e.g., MDIO) needed in ANY low-power state
  - Current designs conservative estimate: 15% of total power
    - Leakage as normal and overhead circuitry is kept at full activity
      - Can be reduced
    - Higher than assumptions in diab_2_0907

- Deep_Sleep_Idle_frame power: P_on= P_nominal – P_LDPC-P_ENX
  - Current designs estimate: 65% nominal power
    - Consistent with overhead+50% analog assumption of diab_2_0907

- 1:N frame decimation gives:
  - P_deep_sleep_idle: [(N-1)*P_off + (P_on)] / N

- 1:10 frame decimation, (equiv 1G traffic load)
  - P= [9*.15+.65] /10 = 20% of nominal 10G PHY power level

Recovery Time

- Bringup time = T_interface sync + T_latency + T_next_frame (+T_ack ?)
  - T_latency (10GBASE-T) = 2.5usec
  - Interface: XAUI: T_bringup = XX usec
  - Time to next frame: T_next_frame can be negotiated
    - Longer times allow deeper power down (leakage & overhead savings)
    - Shorter times allow faster transition, less savings
    - Examples:
      - 1:10 - T_next_active_frame= 3.2usec, 20% nom pwr
      - 1:1000 – T_next_active_frame = 320usec, 8% nom pwr
        - Longer times enable greater overhead savings
- Require Acknowledge of speed transitions?
  - T_ack: time for acknowledge – RECOMMEND AGAINST THIS.
  - Costs time (Latency + processing delay), increases required buffers to account for missed Acks and increases error states

- **ENABLES < 10 usec transitions**

Open issues & advantages

- Negotiation of recovery time (duty cycle)
- Carrying data in “deep_sleep_idle” frames
  - Costs some power, and can be negotiated, complicates MAC/PHY relationship
    - May force negotiation of M “on frames” per N “off frames” (right now M=1)

Deep Sleep Idle Considerations

- Deep Sleep Idle power consumption
    - Can be limited purely by the leakage and overhead management
        - Should always be better than going to the next lower rate or subset
- Transition delays – adjustable, down to microseconds
- Timing recovery -Inherently retains timing lock and frame sync
- Asymmetric operation - possible
    - But more complicated to operate while other direction is deep sleep idle
- Implementation cost & complexity
    - Similar to fast-start, deep sleep idle adds only minimal control sequencing circuitry and doesn't change the signal processing or datapath PCS circuitry of the PHY

Conclusions

- Deep Sleep Idles can be structured by periodic transmission of blocks of data
  - Can use inherent framing in PCS of 10GBASE-T
  - Uses existing PCS and PMA with minimal "flywheel" logic
- 10usec-scale recovery times are achievable
- Achieves adjustable efficiency better than 10X improvement for the PHY
  - Much better for the entire system