Before Meeting with Cole,

1. Discussion about design status – SoCET

* Design Status
  + Integration of PLIC and CLINT with [the bus](https://wiki.itap.purdue.edu/display/ecedesign/PLIC) and/or core.
  + Wrapper for PLIC and CLINT.
  + Modeling/emulating ISR read/write transactions without the core?
    - Refer the TB for cycle level detail (not using core)?
  + Does the chip currently support only [Machine Mode](https://wiki.itap.purdue.edu/display/ecedesign/Privileged+Architecture)? Or does it support U,S modes too now?
  + Vectored and Direct Modes are to be included in test plan? (SW level concern)
* PLIC
  + Are all interrupts [latched](https://github.com/Purdue-SoCET/interrupt_controller/blob/d85961d54c80b84024821226d655ed015792ea82/tb/tb_interrupt_controller.sv#L144)? [How often](https://github.com/Purdue-SoCET/interrupt_controller/blob/d85961d54c80b84024821226d655ed015792ea82/tb/tb_interrupt_controller.sv#L184) are they latched/sampled?
    - Counter for edge triggered intr.
  + Interrupt type – Level or Edge?
  + [Section 1.2.1, 3.2.3](https://sifive.cdn.prismic.io/sifive/d1984d2b-c9b9-4c91-8de0-d68a5e64fa0f_sifive-interrupt-cookbook-v1p2.pdf) - Programmable interrupt levels or priorities?
  + Expected behavior upon receiving multiple interrupts from the same source.
  + [FIXME](https://github.com/Purdue-SoCET/interrupt_controller/blob/d85961d54c80b84024821226d655ed015792ea82/tb/tb_interrupt_controller.sv#L181): Sending in interrupt 7 is not being registered [for some reason](https://wiki.itap.purdue.edu/display/ecedesign/Implementation+Notes)
  + mip CSR bit cleared upon [interrupt claim](https://github.com/Purdue-SoCET/interrupt_controller/blob/d85961d54c80b84024821226d655ed015792ea82/tb/tb_interrupt_controller.sv#L155) or interrupt completion?
  + Does it allow preemption of an in-service routine by a higher priority interrupt?
    - If so, is the context switch handled by the OS? What happens to the CSRs in PLIC (mip)?
    - No
* Can we model a scenario where interrupt request (level triggered only) is cleared/withdrawn from interrupt source? (Should not matter if interrupts are being latched)
  + Before interrupt is claimed
    - After interrupt is claimed (during ISR)
  + What happens when ISR causes an exception? (Highly unlikely)
    - Deep stack of the CSR – 1 excp within an interrupt
  + Claim response
* CLINT
  + Are the resources for timer, timercmp (CSR), duplicated in multi-core systems?
  + [Section 1.2.1, 3.2.3](https://sifive.cdn.prismic.io/sifive/d1984d2b-c9b9-4c91-8de0-d68a5e64fa0f_sifive-interrupt-cookbook-v1p2.pdf) - Fixed priorities?
  + Documentation – SiFive Interrupt Cookbook as reference?
  + Interrupts from both PLIC and CLINT – Resolution through (fixed) priority?
* CLIC – Implementation status
* General/Miscellaneous
  + Does the chip get interrupts from other sources (non-GPIO) on the board?
    - PLIC parameterized for diff num of interrupts
      * Mult of 32
  + Bus Mux
  + Interleaving of Interrupts - UART,USB,GPIO, etc. Need to save additional states? Handler takes care of this?

1. Design specs and usage

* GPIO
* Timer

1. Test Plan
2. Design Logs - Ruoyi Chen

DUT compile with just the top level TB

After Meeting with Cole,

Discussion about design status – SoCET

* Design Status
  + Integration of PLIC and CLINT with [the bus](https://wiki.itap.purdue.edu/display/ecedesign/PLIC) and/or core.
    - PLIC and CLINT wrappers (integration with AHB) can be found in src.
  + Should the TB model/emulate ISR read/write transactions without the core?
    - No, that would not be necessary.
  + Should I go through the design (RTL) to figure out cycle level detail (not using core)?
    - The reads and writes to interrupt claim/complete register should be enough to orchestrate the DUT. Cycle-accurate details not required.
  + Does the chip currently support only [Machine Mode](https://wiki.itap.purdue.edu/display/ecedesign/Privileged+Architecture)? Or does it support U,S modes too now?
    - For now, only machine mode is supported.
    - U mode is in plan for future.
  + Vectored and Direct Modes are to be included in test plan?
    - No, they are SW/kernel level concern
* PLIC
  + Are all interrupts [latched](https://github.com/Purdue-SoCET/interrupt_controller/blob/d85961d54c80b84024821226d655ed015792ea82/tb/tb_interrupt_controller.sv#L144)? [How often](https://github.com/Purdue-SoCET/interrupt_controller/blob/d85961d54c80b84024821226d655ed015792ea82/tb/tb_interrupt_controller.sv#L184) are they latched/sampled?
    - Yes. GPIO unit detects edge sensitive interrupts and sends level sensitive interrupts to interrupt controller.
    - Counter for edge triggered intr?
      * No
  + Interrupt type – Level or Edge?
    - GPIO – Edge
    - Interrupt Controller – Level (and is latched)
  + [Section 1.2.1, 3.2.3](https://sifive.cdn.prismic.io/sifive/d1984d2b-c9b9-4c91-8de0-d68a5e64fa0f_sifive-interrupt-cookbook-v1p2.pdf) - Programmable interrupt levels or priorities?
    - Part of CLIC. Not CLINT, PLIC.
  + Expected behavior upon receiving multiple interrupts from the same source?
    - For now, there’s no support for interrupt edge counters (GPIO).
    - So, unless an interrupt source is cleared, you cannot send another interrupt from that source.
  + [FIXME](https://github.com/Purdue-SoCET/interrupt_controller/blob/d85961d54c80b84024821226d655ed015792ea82/tb/tb_interrupt_controller.sv#L181): Sending in interrupt 7 is not being registered [for some reason](https://wiki.itap.purdue.edu/display/ecedesign/Implementation+Notes)
    - Cole did not encounter this issue.
  + mip CSR bit cleared upon [interrupt claim](https://github.com/Purdue-SoCET/interrupt_controller/blob/d85961d54c80b84024821226d655ed015792ea82/tb/tb_interrupt_controller.sv#L155) or interrupt completion?
    - When interrupt comes in, the ID of the interrupt will be reflected in interrupt Claim/Complete register.
    - When this register is read from, an interrupt claim is performed. PLIC will internally clear the corresponding interrupt pending bit from mip reg.
    - Interrupt service is completed when a write is performed to the interrupt Claim/Complete register.
    - PLIC moves on to the next interrupt source only after interrupt completion. So PLIC will send a new interrupt request pulse if,
      * There’s a new interrupt or,
      * If an interrupt has been serviced/completed and there’s another pending interrupt.
  + Does it allow preemption of an in-service routine by a higher priority interrupt? If so, is the context switch handled by the OS? What happens to the CSRs in PLIC (mip)?
    - No, preemption is not supported.
* Can we model a scenario where interrupt request (level triggered only) is cleared/withdrawn from interrupt source? (Should not matter if interrupts are being latched)
  + Before interrupt is claimed
    - After interrupt is claimed (during ISR)
    - No once an interrupt (edge) is detected and latched, it cannot be withdrawn.
  + What happens when ISR causes an exception? (Highly unlikely)
    - Deep stack of the CSR – 1 excp within an interrupt
    - SW/Kernel level concern.
* CLINT
  + Are the resources for timer, timercmp (CSR), duplicated in multi-core systems?
    - Timercmp should be separate for each core. For now, we have only single core.
  + [Section 1.2.1, 3.2.3](https://sifive.cdn.prismic.io/sifive/d1984d2b-c9b9-4c91-8de0-d68a5e64fa0f_sifive-interrupt-cookbook-v1p2.pdf) - Fixed priorities?
    - This concerns CLIC.
  + Use SiFive’s Interrupt Cookbook as reference?
    - Yes
  + Interrupts from both PLIC and CLINT – Resolution through (fixed) priority?
* CLIC – Implementation status
  + No plans as of yet
* General/Miscellaneous
  + Does the chip get interrupts from other sources (non-GPIO) on the board?
  + PLIC parameterized for diff num of interrupts
    - Mult of 32
    - Non multiples of 32.
  + Interleaving of Interrupts - UART,USB,GPIO, etc. Need to save additional states? Handler takes care of this?
    - All the inputs go through the GPIOs, be it USB, UART, SPI. So if PLIC works fine, the HW doesn’t have to worry about this.

1. Design specs and usage

* GPIO
* Timer

1. Test Plan – Eliminated some cases. Scope reduced
2. Design Logs - Ruoyi Chen