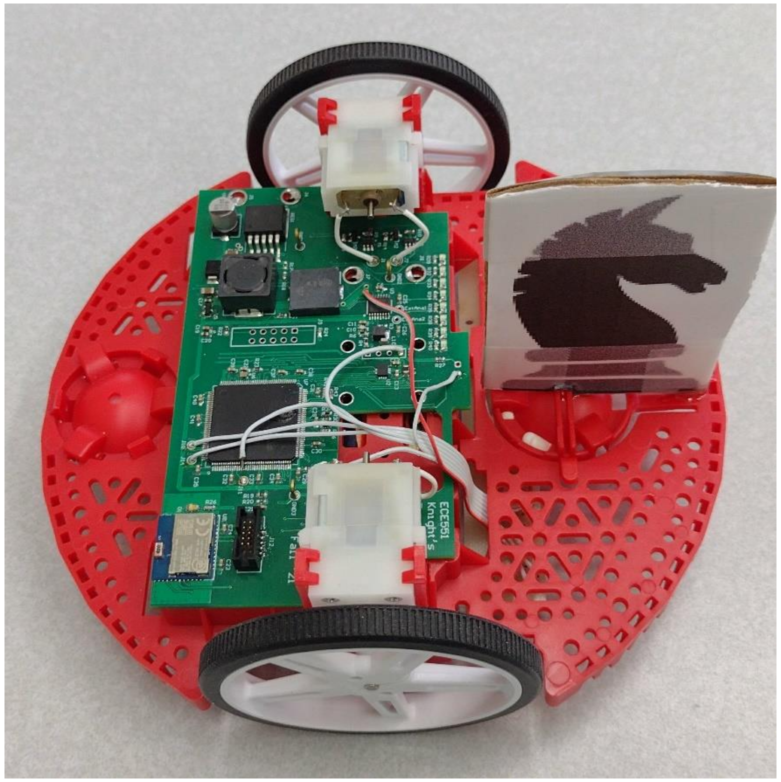
ECE 551
Project Spec

Fall ‘21

Knight’s Tour



Grading Criteria: (Project is 28% of final grade)

- Project Grading Criteria:
  - Quantitative Element 15%

    (yes this could result in extra credit)
  - Project Demo (85%)
    - ✓ Code Review (12.5%)
    - ✓ Testbench Method/Completeness (17.5%)
    - ✓ Synthesis Script review (7.5%)
    - ✓ Post-synthesis Test run results (10%)
    - ✓ Results when placed in EricTommy Testbench (22.5%)
    - ✓ Run of "the knight" on the board (15%)

$$Quantitative = \left(\frac{EricTommys\_ProjectArea}{YourSynthesizedArea}\right)^{0.75}$$

Note: The design has to be functionally correct for this to apply

<u>Extra Credit Opportunity:</u>

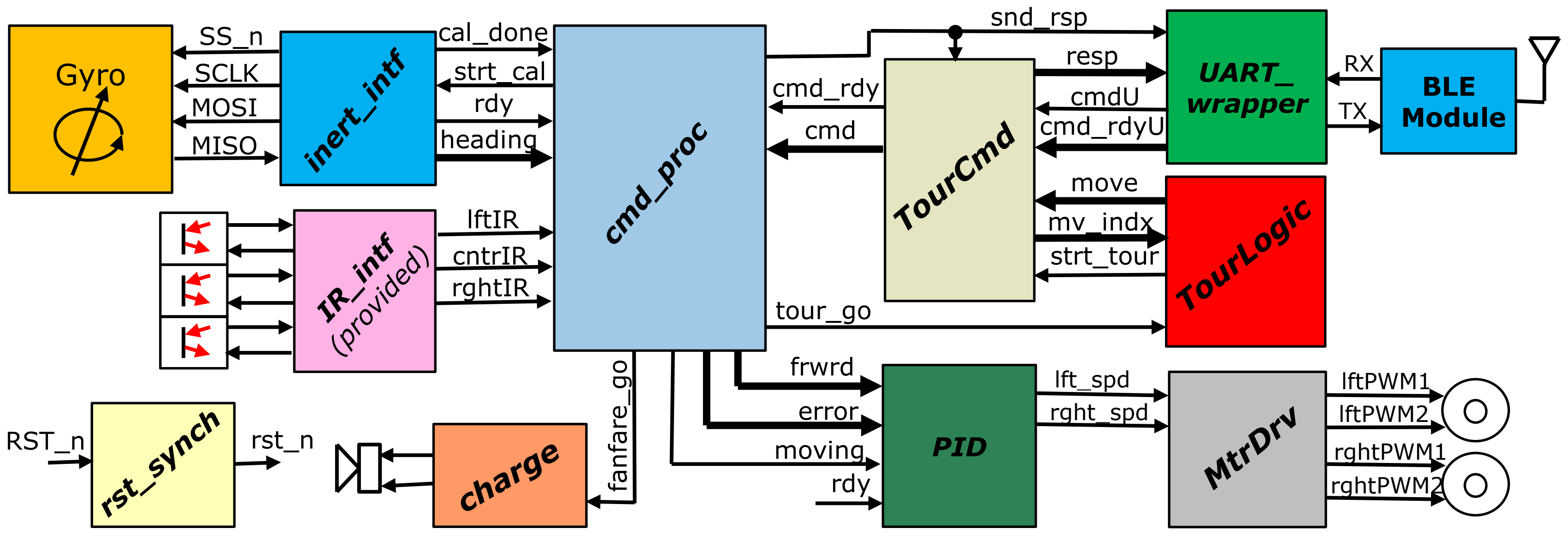
Appendix C of ModelSim tutorial instructs you how to run code coverage (NOTE: this only works on vsim on the linux machines)

- Run code coverage on a single test and get 1% extra credit
- Run code coverage across your test suite and get a cumulative number and get 2% extra credit.
- Run code coverage across your test suite and give **concrete** example of how you used the results to improve your test suite and get 3% extra credit.

Project Due Date

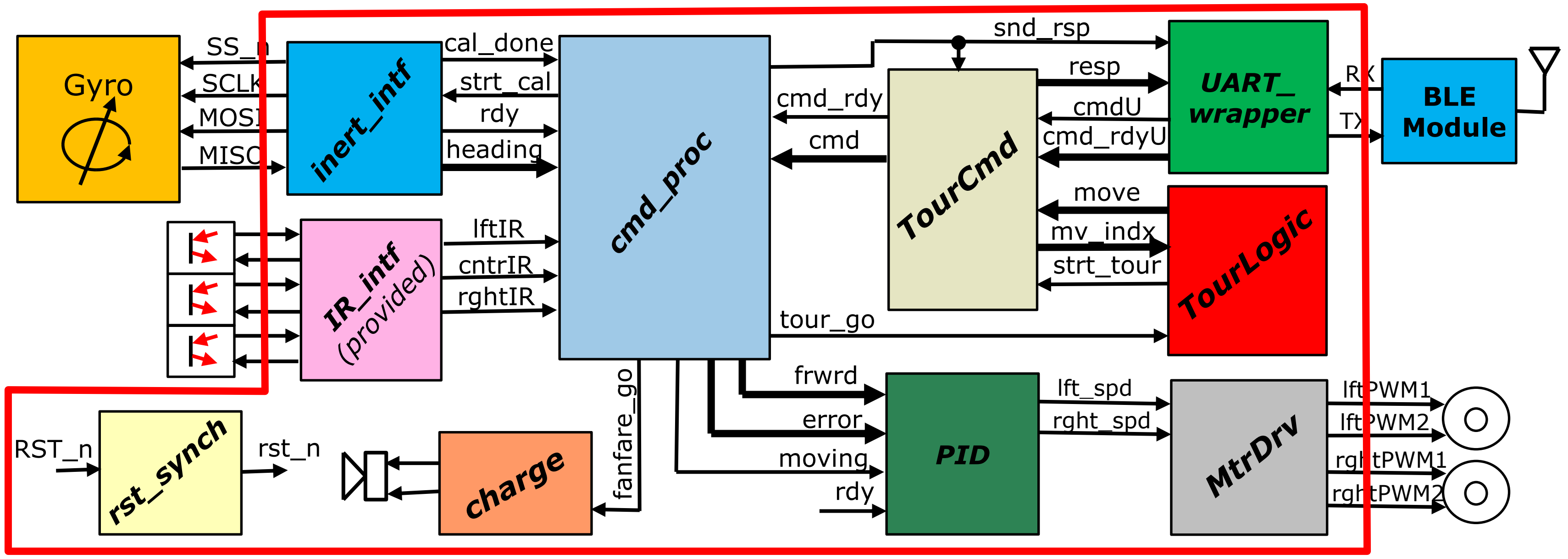
- Project Demo Involves:
  - ✓Code Review
  - ✓ Testbench Method/Completeness
  - ✓ Synthesis Script & Results review
  - ✓ Post-synthesis Test run results
  - ✓ Results when placed in EricTommy testbench
  - ✓ Results when mapped to FPGA and run on the chess board

Block Diagram of Digital Portion



Show above is the arrangement of all the blocks you have completed during the course of the semester. This block diagram gives you a rough idea how they all work together.

Block Diagram of Digital Portion



The blocks outlined in red above are pure digital blocks, and will be coded with the intent of being synthesized via Synopsys to our standard cell library. For practical purposes we will also map that logic to a DE0 FPGA board so we can run the demos.

You Must have a block called **KnightsTour.sv** which is top level of what will be the synthesized DUT. A shell is provided to ensure we all have the same interface.

The hierarchy/partitioning of your design below **KnightsTour.sv** is up to your team. The hierarchy of your testbench is up to your team.

KnightsTour Interface

Signal Name:	**Dir:**	**Description:**
clk	in	Clock input (50MHz)
RST_n	in	Active low input from push button. Should be synchronized inside follower to produce **rst_n** which goes to all other units as the global reset.
MISO	in	Monarch In Serf Out from SPI bus of MEMs gyro
SS_n	out	Active low serf select to gyro
SCLK	out	SPI bus clock (to gyro)
MOSI	out	Monarch Out Serf In (to gyro)
INT	in	New set of reading ready from gyro
RX	in	UART data in from Bluetooth module (cmd)
TX	out	UART data out to Bluetooth module (resp)
IR_en	out	Enables emitter on the 3 IR sensors
lftIR_n, cntrIR_n, rghtIR_n	in	Raw inputs from IR sensors. Conditiond by IR_intf *(provided block)* to produce lftIR, cntrIR, rghtIR
piezo, piezo_n	out	Drives to piezo buzzer for charge fanfare
lftPWM1,lftPWM2	out	PWM signals to control motor drive magnitude
rghtPWM1,rghtPMW2	out	Motor direction signals 1 => reverse 0 => forward

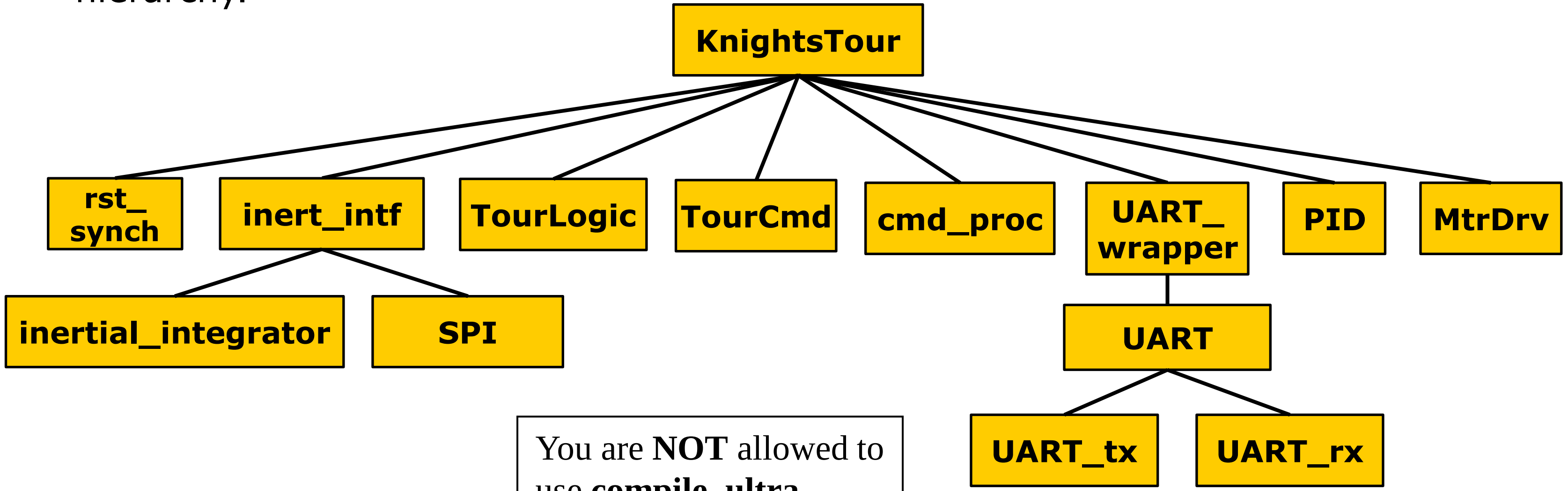
Provided Modules & Files: (available on website under: Project)

File Name:	Description:
KnightsTour_tb.sv	**Optional** testbench template file.
KnightPhysics.sv	Very helpful model of the physics of “the Knight” and the board. Use this in conjunction with your testbench to validate behavior.
KnightsTour.sv	**Requried** interface skeleton verilog file. **Copy this** and flush it out with your design
SPI_iNEMO4.sv	Final model of NEMO inertial sensor (gyro) intended for use along with KnightPhysics.
inertial_integrator.sv	**NOTE**: this is slightly different than the one you were given earlier in Ex19. This one has some pipeline flops added to help synthesis speed paths.
IR_intf.sv	Interface to inertial sensors. Produces lftIR, cntrIR, rghtIR

These files exist in: **KnightsTourProvidedFiles.zip**

Synthesis:

- You have to be able to synthesize your design at the **KnightsTour** level of hierarchy.



You are **NOT** allowed to use **compile_ultra**

- Your synthesis script should write out a gate level netlist of follower (**KnightsTour.vg**).
- You should be able to demonstrate at least one of your tests running on this post synthesis netlist successfully.
- Timing (350MHz) (2.857ns) is mildly challenging. Your main objective is to minimize area.

Synthesis Constraints:

Contraint:	Value:
Clock frequency	333MHz (yes, I know the project spec speaks of 50MHz, but that is for the FPGA mapped version. The standard cell mapped version needs to hit 333MHz. 3ns period
Input delay	0.4ns after clock rise for all inputs
Output delay	0.4ns prior to next clock rise for all outputs
Drive strength of inputs	Equivalent to a size 2, 2-input NAND gate from our library
Output load	0.1pF on all outputs
Wireload model	16000
Max transition time	0.15ns
Clock uncertainty	0.15ns

NOTE: Area should be taken after all hierarchy in the design has been smashed.

Eric's Area = 16242

Tommys's Area = 14080

AreaUsedInFormula = 15161

Project Demos (Held in "The Dungeon" B555)

- Project Demos will be held in B555 over the course of 4 days.
  - Friday Dec 10th. 3% Extra credit for demoing very early
  - Monday Dec 13th 1.5% Extra credit for demoing early
  - Tuesday Dec 14th 1% Extra credit for demoing bit early
  - Wednesday Dec 15th   Normal due time
  - Thursday Dec 16th 0.5% Penalty for demoing late

  - Project Demo (85%)   (15% is based on your synthesize area)
    - ✓ Code Review (12.5%)
    - ✓ Testbench Method/Completeness (17.5%)
    - ✓ Synthesis Script review (7.5%)
    - ✓ Post-synthesis Test run results (10%)
      - ❑ You need to demonstrate post synthesis simulation of a short test at the "KnightsTour" level of hierarchy.
    - ✓ Results when placed in EricTommy Testbench (22.5%)
    - ✓ Run of "the knight" on the board (15%)