UNITED STATES AIR FORCE ACADEMY

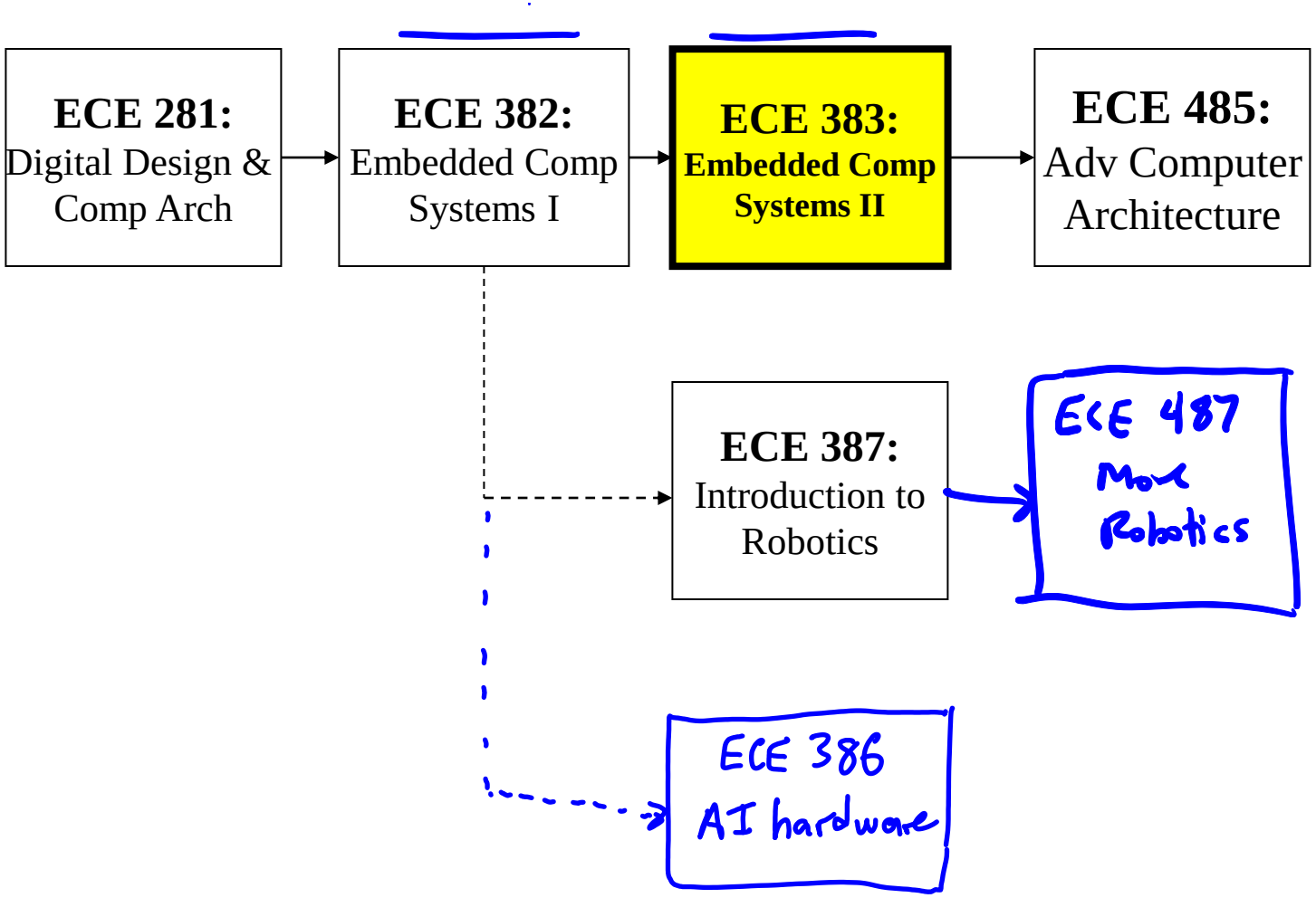
- external drives → copy vivado folder
- FPGA Boards
  -- 1 Board, 1 power supply, 2 USB cables
  2 aux cables

UNITED STATES AIR FORCE ACADEMY

ECE 383 – Embedded Computer Systems II Lecture 1 – Intro to Digital System Design

Dr George York
Room 2E46E
333-4210
719-484-9608

DFEC Computer Engineering Courses



UNITED STATES AIR FORCE ACADEMY

Lesson Outline

1. Why digital systems?
2. Instructor/Course Introduction
3. Methods of implementing digital systems
4. Custom digital device technologies
5. Abstraction
6. Digital System Design
7. Design Goals
8. Digital Design – Majority Circuit

Vivado version 2018.2

HW #1 → install Vivado → Bitbucket
→ answer questions in GradeScope

INSTRUCTOR/COURSE INTRO

Dr George York

- Education (embedded computers & DSP)
  - BSEE, USAFA '86
  - MSEE & PhD, U of Washington, Seattle
- USAF Career, 62E
  - Eglin AFB (AFRL Munitions)-- smart bombs
  - Taejon, Korea (Engr/Scientist Exchange Program)
  - USAF Academy, '92-'94, '02-'04, '09-now
  - PhD – Medical Ultrasound Machines
  - National Security Agency (NSA), Tactical SIGINT
  - London, UK (AFOSR/EOARD)
- My Spare Time
  - Cycling Team

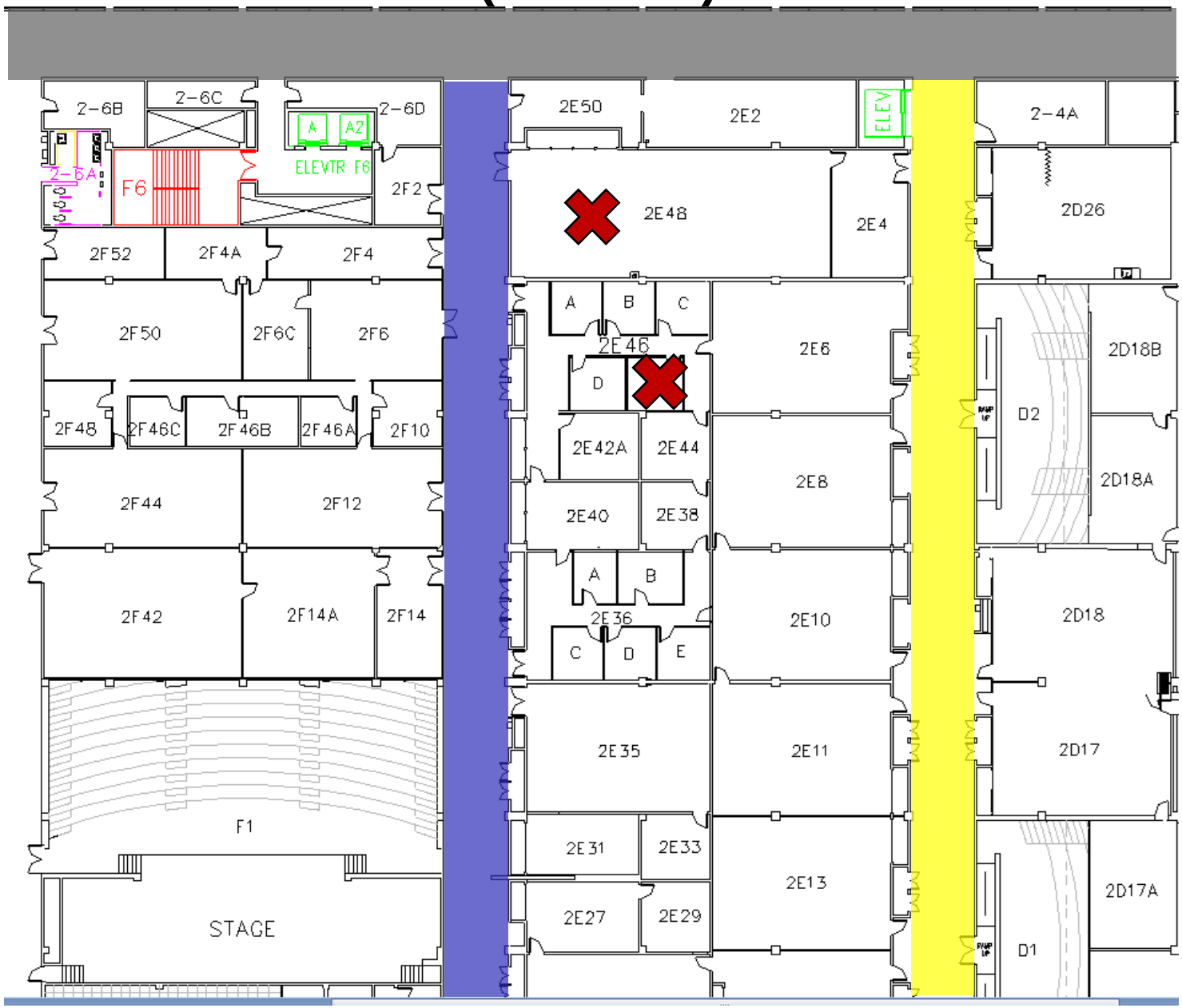
UNITED STATES AIR FORCE ACADEMY

Where is Dr York?

Period	M-Day	T-Day
1		ECE 485
2		
3		Capstone
4		Capstone
5		
6	ECE 383	
7	ECE 383	

Book with Me

West (Terrazzo) Side



UNITED STATES AIR FORCE ACADEMY

Rules

1. **Be prepared for class (reading/homework)**
2. **Pay attention in lecture (1st hour)**
3. **Be productive during 2nd hour of application**

Final Exam?

UNITED STATES AIR FORCE ACADEMY

Course Materials/etc

- **Website:**
  - https://georgeyork.github.io/ECE383_web
- **Bitbucket and Gradescope**
- **Textbook:**
  - RTL Hardware Design Using VHDL, Pong P. Chu
  - I have found it free via the Air Force in the AF e-Learning environment.
  - In the AF Portal under Career & Training, click on AF e-Learning.
  - From there I clicked on Browse The Library at the top and entered RTL in the Search box at the top of the page.
  - Your requested title is second in my results list.

UNITED STATES AIR FORCE ACADEMY

Lab Reports

Note on Lab Reports:

- **Last year created README in repo (mark down), and grade feedback with bitbucket's "issue tracker"**
- **This year? Maybe provide shell of lab report in Word, and submit/feedback via gradescope?**

WHY DIGITAL SYSTEMS?

UNITED STATES AIR FORCE ACADEMY

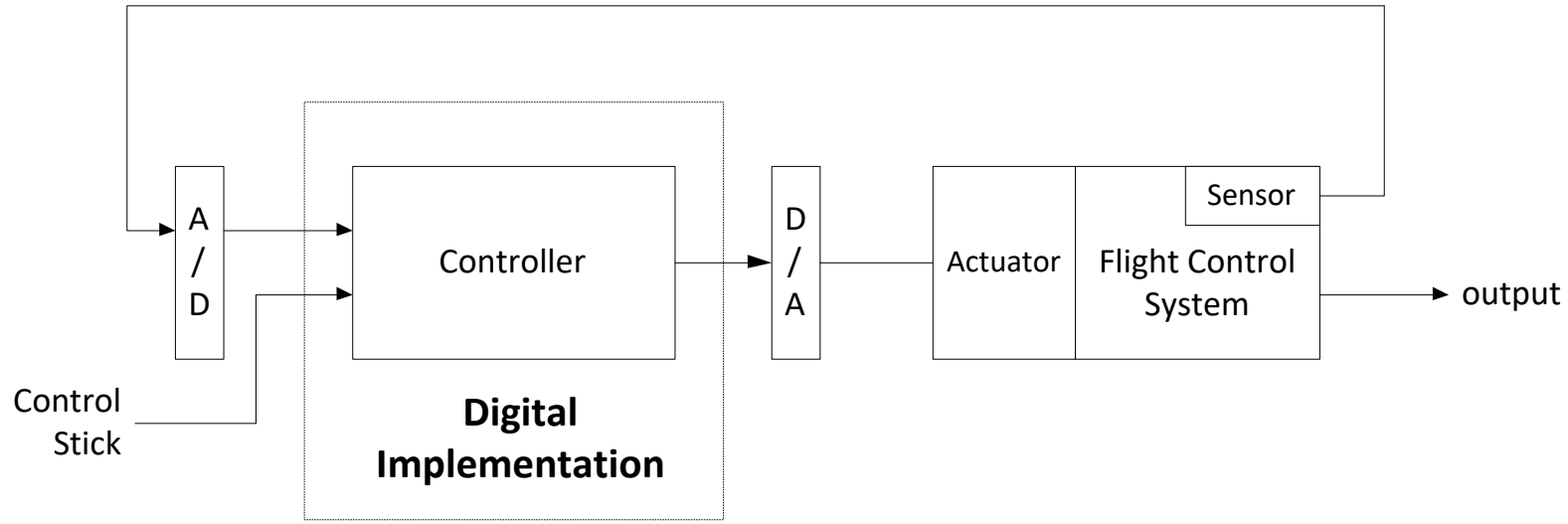
Why Digital Systems?

vs Analog?

- **Advantages**
  - **Reproducibility of information**
  - **Flexibility and functionality: easier to store, transmit and manipulate information**
  - **Economy: cheaper device and easier to design**
- **Digital circuitry replaces many analog systems:**
  - ***Audio recording:*** **from tape to music CD to MP3 (MPEG Layer 3) player**
  - ***Image processing:*** **from silver-halide film to digital camera**
  - **Telephone switching networks**
  - **Control of mechanical system: e.g., "fly-by-wire"**

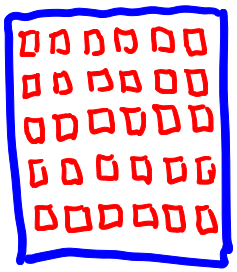
First fly-by-wire Fighter?

Why Digital Systems?



Fly-By-Wire Digital System

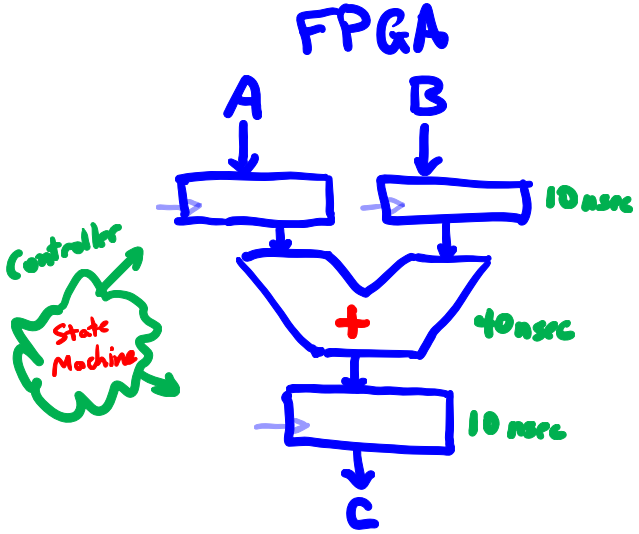
METHODS OF IMPLEMENTING DIGITAL SYSTEMS



FPGA vs Microcontroller?

- What is an FPGA? vs ASIC?
- What is a Microprocessor? (or computer?)
- Which is faster?

Task: $C = A \cdot B$



Microprocessor

8000 LD R2, A
8004 LD R3, B
8008 MUL R1, R2, R3*
800C ST R1, C

Per Instruction
F 40nsec
D 40nsec
E 40nsec
S 40nsec

total time
$4 \cdot 4 \cdot 40\text{nsec} = 640_{\text{nsec}}$

UNITED STATES AIR FORCE ACADEMY

Digital Implementation Methods

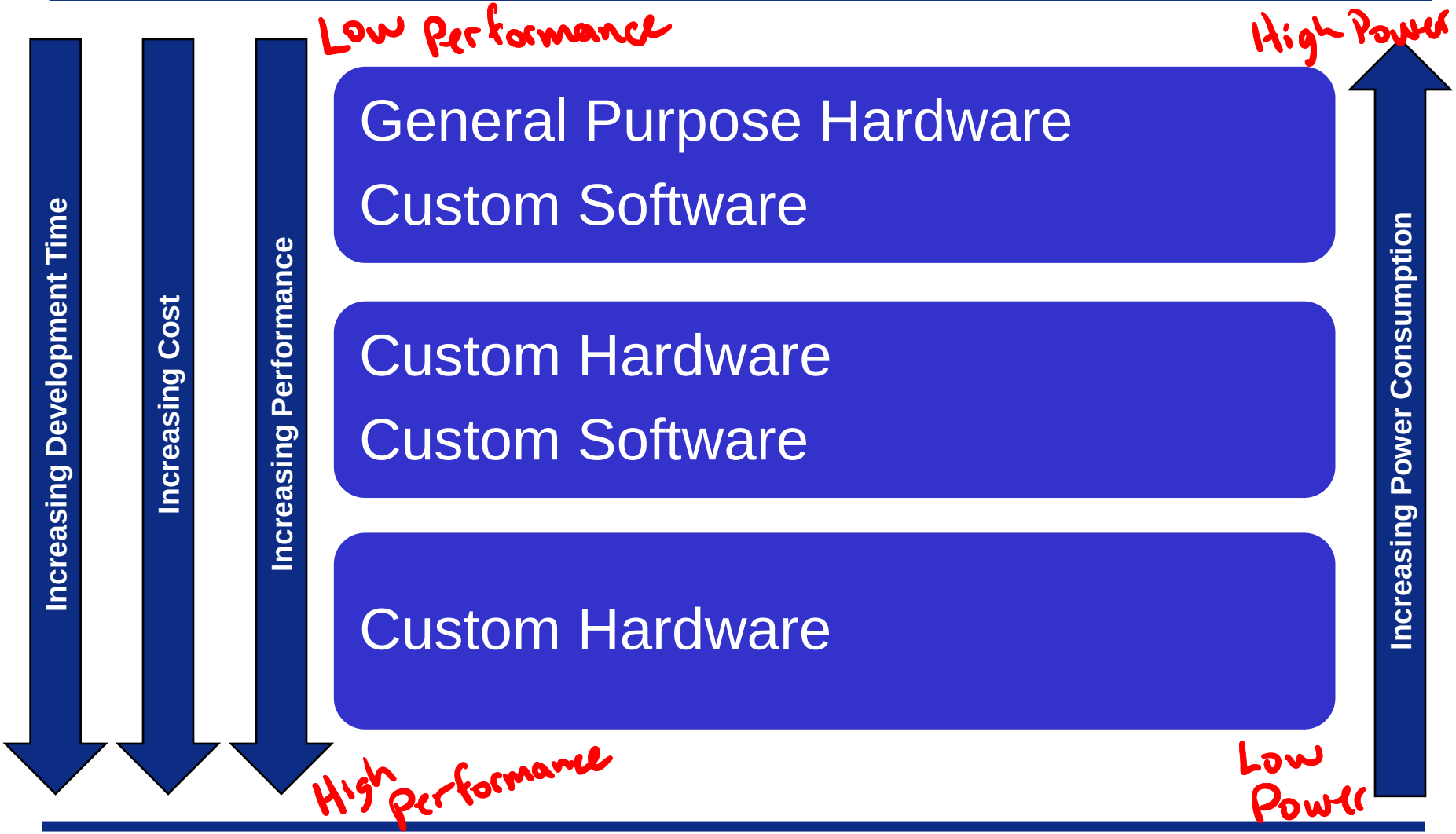
382?

1. **General-purpose hardware with custom software**
   - **General purpose processor**
     - **Performance-oriented processor (e.g., Core i7)**
     - **Cost-oriented processor (e.g., PIC microcontroller)**
   - **Special purpose processor**
     - **DSP processor (multiplication-addition)**
     - **Network processor (buffering and routing)**
     - **Graphics engine (3D rendering)**
2. **Custom software on a custom processor (known as hardware-software co-design)**
3. **Custom hardware**

Note: A complex project may use more than one of these!

UNITED STATES AIR FORCE ACADEMY

Digital Implementation Methods



CUSTOM DIGITAL DEVICE TECHNOLOGIES

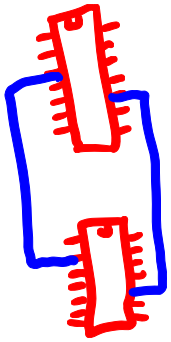
UNITED STATES AIR FORCE ACADEMY

Custom Digital Device Technologies

VLSI

ECE 373 473

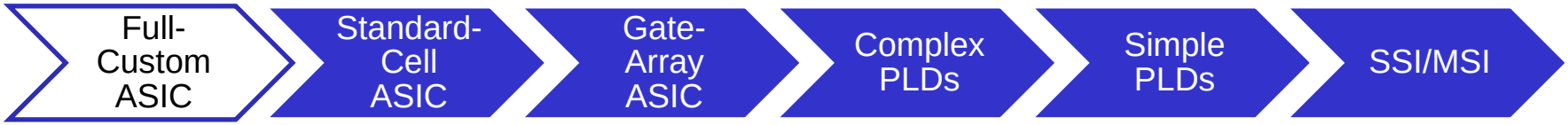
- **Where customization is done:**
  - **In a fab (fabrication facility): ASIC (Application Specific IC)**
  - **In the "field": non-ASIC**
- **Six device technology classifications:**



1. **Full-custom ASIC**
2. **Standard cell ASIC**
3. **Gate array ASIC**
4. **Complex field programmable logic device**
5. **Simple field programmable logic device**
6. **Off-the-shelf SSI (Small Scaled IC)/MSI (Medium Scaled IC) components** LSI

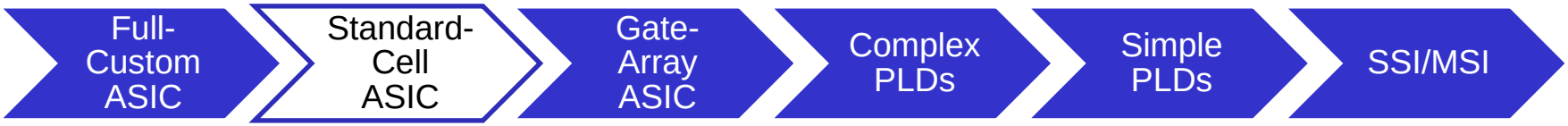
Full-custom ASIC

- All aspects (e.g., size of a transistor) of a circuit are tailored for a particular application.
- Circuit fully optimized
- Design extremely complex and involved
- Only feasible for small components
- Masks needed for all layers



Standard-Cell ASIC

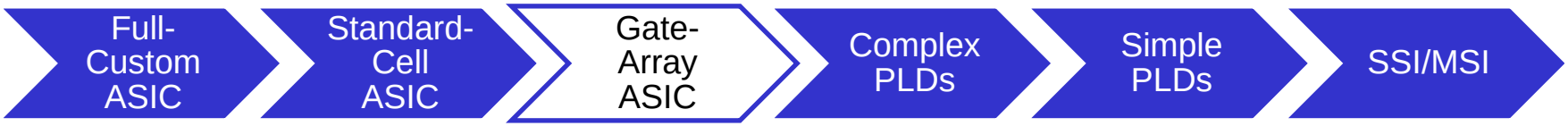
- Circuit made of a set of pre-defined logic, known as *standard cells*
  - Basic logic gates
  - 1-bit adder,
  - D FF
  - etc.
- Layout of a cell is pre-determined, but layout of the complete circuit is customized
- Masks needed for all layers



UNITED STATES AIR FORCE ACADEMY

Gate array ASIC

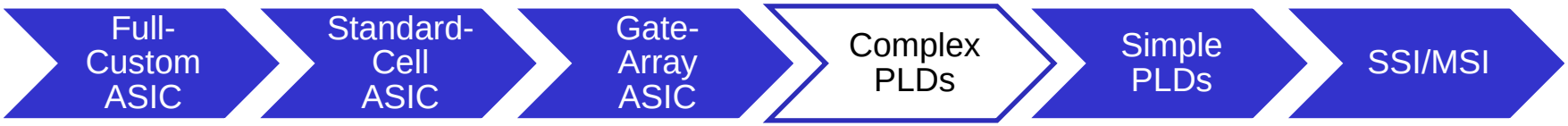
- Circuit is built from an array of a single type of cell (known as *base cell*)
- Base cells are pre-arranged and placed in fixed positions, aligned as one- or two-dimensional array
- More sophisticated components (*macro cells*) can be constructed from base cells
- Masks needed only for metal layers (connection wires)



UNITED STATES AIR FORCE ACADEMY

Complex Field Programmable Device

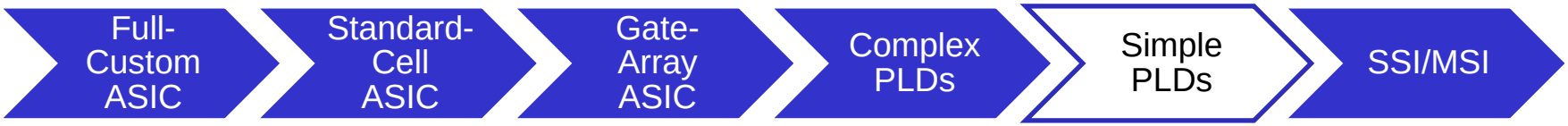
- Device consists of an array of generic logic cells and general interconnect structure
- Logic cells and interconnect can be "programmed" by utilizing semiconductor *fuses* or *switches*
- Customization is done "in the field" vs. fab
- Two categories:
  - CPLD (Complex Programmable Logic Device)
  - FPGA (Field Programmable Gate Array)
- No custom mask needed



UNITED STATES AIR FORCE ACADEMY

Simple Field Programmable Device

- Programmable device with simple internal structure
  - PROM (Programmable Read Only Memory)
  - PAL (Programmable Array Logic)
  - etc.
- No custom mask needed
- Replaced by CPLD/FPGA



UNITED STATES AIR FORCE ACADEMY

1970's

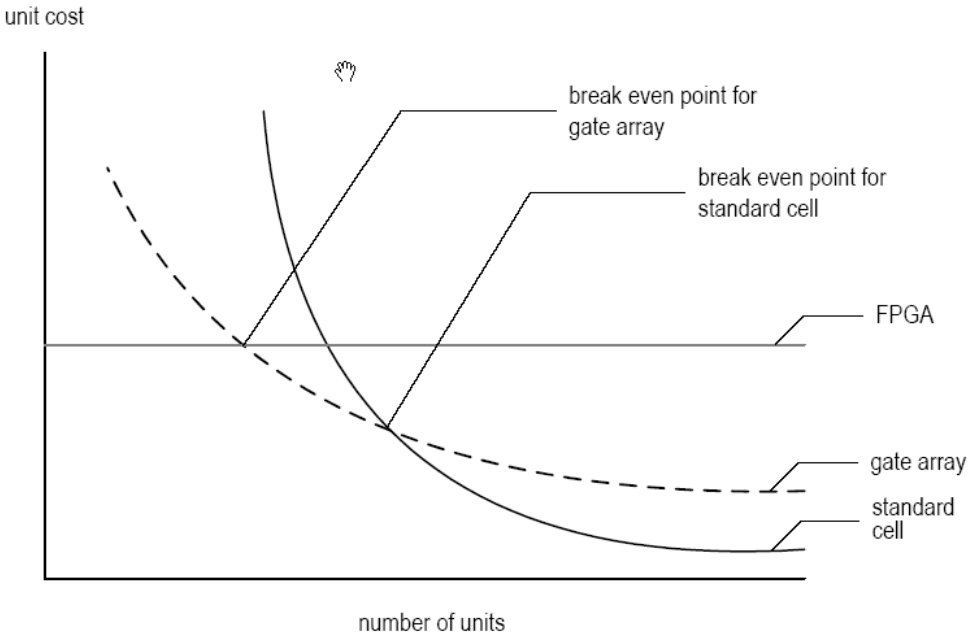
SSI/MSI components

- Small parts with fixed, limited functionality
  - 7400 TTL series (more than 100 parts)
  - etc.
- Resource is consumed by *package* but not *silicon*:
  - Power
  - Board area
  - Manufacturing cost
  - etc.
- No longer a viable option

Automobile NRE?

Types of Cost

- ***NRE (Non-Recurrent Engineering) cost*: one-time, per-design cost**
- ***Part cost*: per-unit cost**
- ***Time-to-market cost*: loss of revenue**



$$C_{per\_unit} = C_{per\_part} + \frac{C_{nre}}{\text{units produced}}$$

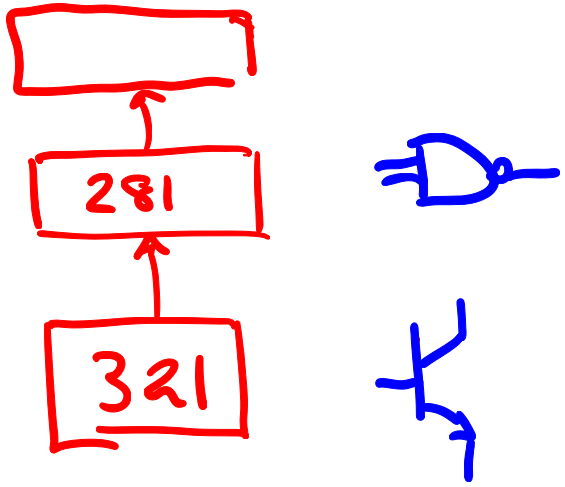
UNITED STATES AIR FORCE ACADEMY

Technology Summary

- **Trade-off between optimal use of hardware resource and design effort/cost**
- **No single best technology**

ASIC

	FPGA	Gate array	Standard cell
tailored masks	0	3 to 5	15 or more
area			best (smallest)
speed			best (fastest)
power			best (minimal)
NRE cost	best (smallest)		
per part cost			best (smallest)
design cost	best (easiest)		
time to market	best (shortest)		
per unit cost		depend on volume	

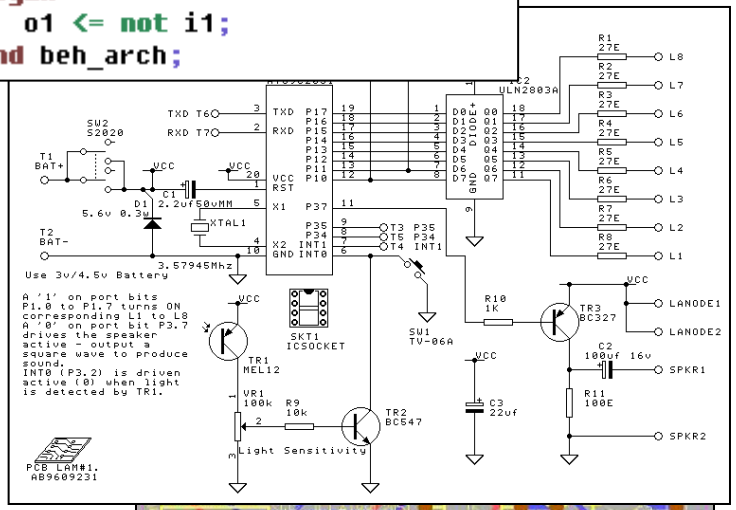


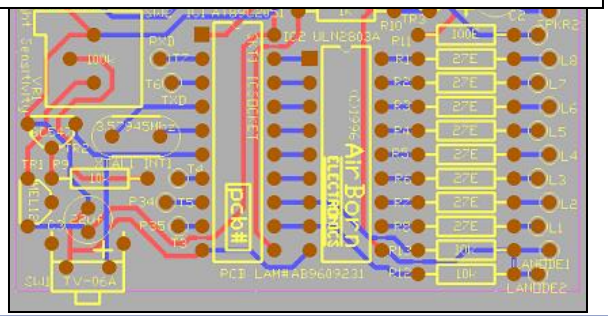
ABSTRACTION

Digital System Views

- **Behavioral view:**
  - **Describe functionalities and I/O behavior**
  - **Treat the system as a black box**
- **Structural view:**
  - **Describe the internal implementation (components and interconnections)**
  - **Essentially block diagram**
- **Physical view:**
  - **Add more info to structural view: component size, component locations, routing wires**
  - **E.g., layout of a print circuit board**

```
    port(
       i1: in std_logic;
       o1: out std_logic
    );
end not1;
architecture beh_arch of not1 is
begin
    o1 <= not i1;
end beh_arch;
```

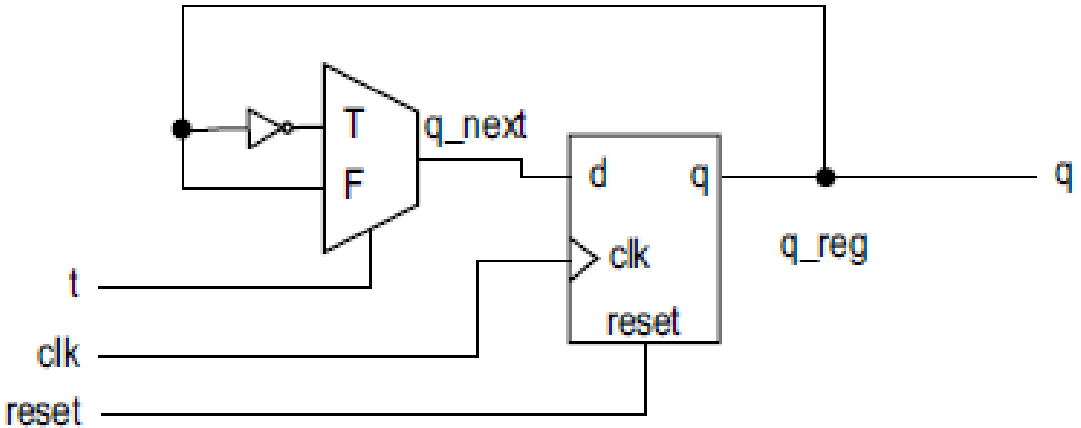




Register-Transfer Abstraction

RTL BBB

- Contains higher-level components (register, adder, mux, etc.) – think of datapaths in ECE 281/382
- Based on clock "tick" event
- Described as a finite state machine
- Later on: a *design methodology* in which the system operation is described by how the data is manipulated and moved among registers



DIGITAL SYSTEM DESIGN

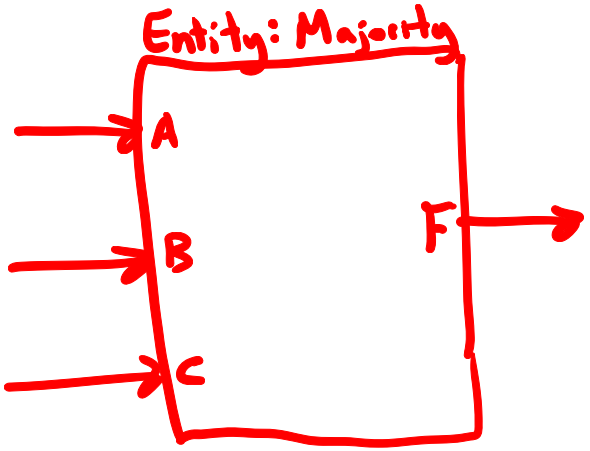
VHSIC Hardware Design Language

VHDL Design

Verilog

- VHDL is just a language which is used to describe hardware circuits.
- A piece of hardware is described in VHDL in two separate ways.
  - Entity – Describes the inputs and outputs
  - Architecture – Describes what transformation the box performs.
- There are two good reasons to realize a design in VHDL, you can
  - simulate the hardware
  - synthesize the hardware

"C" Prototype



UNITED STATES AIR FORCE ACADEMY

Digital System Simulation

- **Simulation** TestBench
  - **When a design is simulated you have complete control of time and the values of all the signals (wires) in the design.**
  - **Aids in Debugging**
  - **We will use Xilinx Vivado to perform our VHDL simulations.**

UNITED STATES AIR FORCE ACADEMY

Digital System Implementation

VHDL

1. **Synthesis** BBB
   - Maps a higher-level description to lower-level components (RT, gate, technology map levels)
   - Results in structural view
2. **Physical Design** Implementation
   - Generates netlist based on synthesis
   - Floor Plan – layout based on RT/processor level
   - Place & Route – gate level
   - Circuit Extraction – Compute propagation delays (Cp/R)
   - Power/Clock Networks
   - Etc.

↘ .bit file

Digital System Implementation

3. **Verification – Checking whether a design meets the functional and timing goals**
   - **Simulation**
   - **Formal verification**
   - **Hardware emulation**
4. ***Testing* – Process of detecting physical defects of a die or package that occurred during manufacturing**

DESIGN GOALS

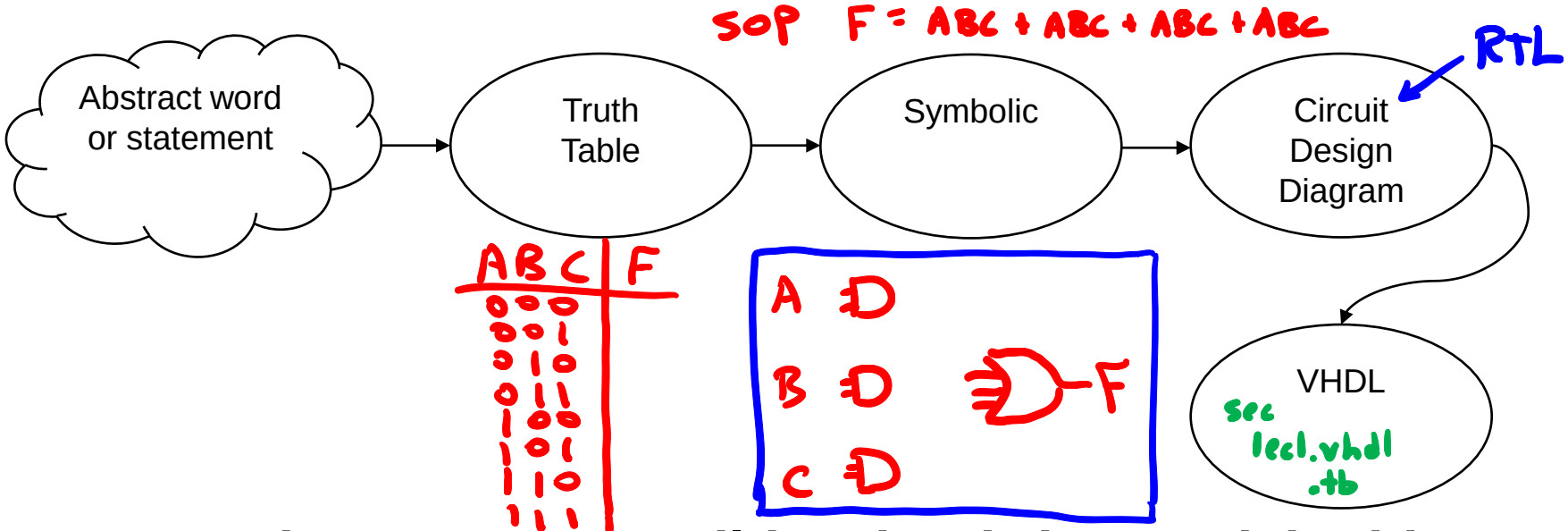
UNITED STATES AIR FORCE ACADEMY

Design Goals

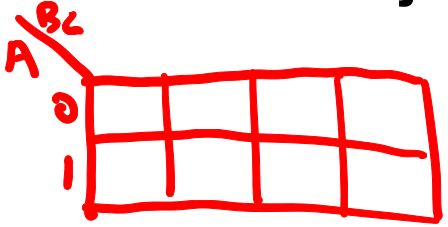
1. **Design for Efficiency**
   - **Synthesis cannot convert bad designs into good ones**
   - **Know what hardware your HDL will create**
2. **Design for Large**
   - **Design a large module**
   - **Design to be incorporated into a larger system**
   - **Design to facilitate the overall development process**
3. **Design for Portability**
   - **Device independent**
   - **Software independent**
   - **Design reuse**

DIGITAL DESIGN – MAJORITY CIRCUIT

Digital Design – Majority Circuit



- **Word Statement:** **Build a circuit (SOP min) with 3-bits of input and 1-bit of output. The output equals 1 when a majority of the three inputs equal 1.**



SOP_{min} =

UNITED STATES AIR FORCE ACADEMY

Important Notes

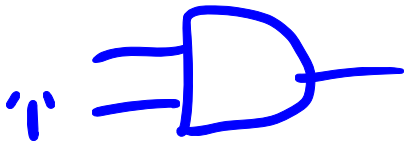
- You can use all the standard logic gates including:
  - AND
  - OR
  - XOR
  - NAND
  - NOR
  - XNOR
  - NOT

what value can a signal have of type std_logic?

UNITED STATES AIR FORCE ACADEMY

Important Notes

- The "data type" std_logic can represent much more than just a logic 0 or 1.
  - 'U', -- Uninitialized
  - 'X', -- Forcing Unknown
  - '0', -- Forcing 0
  - '1', -- Forcing 1
  - 'Z', -- High Impedance
  - 'W', -- Weak Unknown
  - 'L', -- Weak 0
  - 'H', -- Weak 1
  - '-' -- Don't care



UNITED STATES AIR FORCE ACADEMY

Vivado, etc

- **Got Vivado Installed?**
- **BitBucket:**
  - **Create Repo for ECE383**
  - **Give me access: ~~GeorgeYork~~** george.yorke@usafa.edu
    - Settings→user & group access →add GeorgeYork, read access
  - **Issue Tracker**
    - Settings→Issue Tracker→Private Issue Tracker

- **Boards:**
  - **Digilent→Artix-7→Nexys Video** (xc7a200tsbg484-1)

- **If testbench comes up as source file**
  - **Move to simulation sources**

UNITED STATES AIR FORCE ACADEMY

HW#1, due BOC T2

- Turn in via Gradescope, and post code to bitbucket