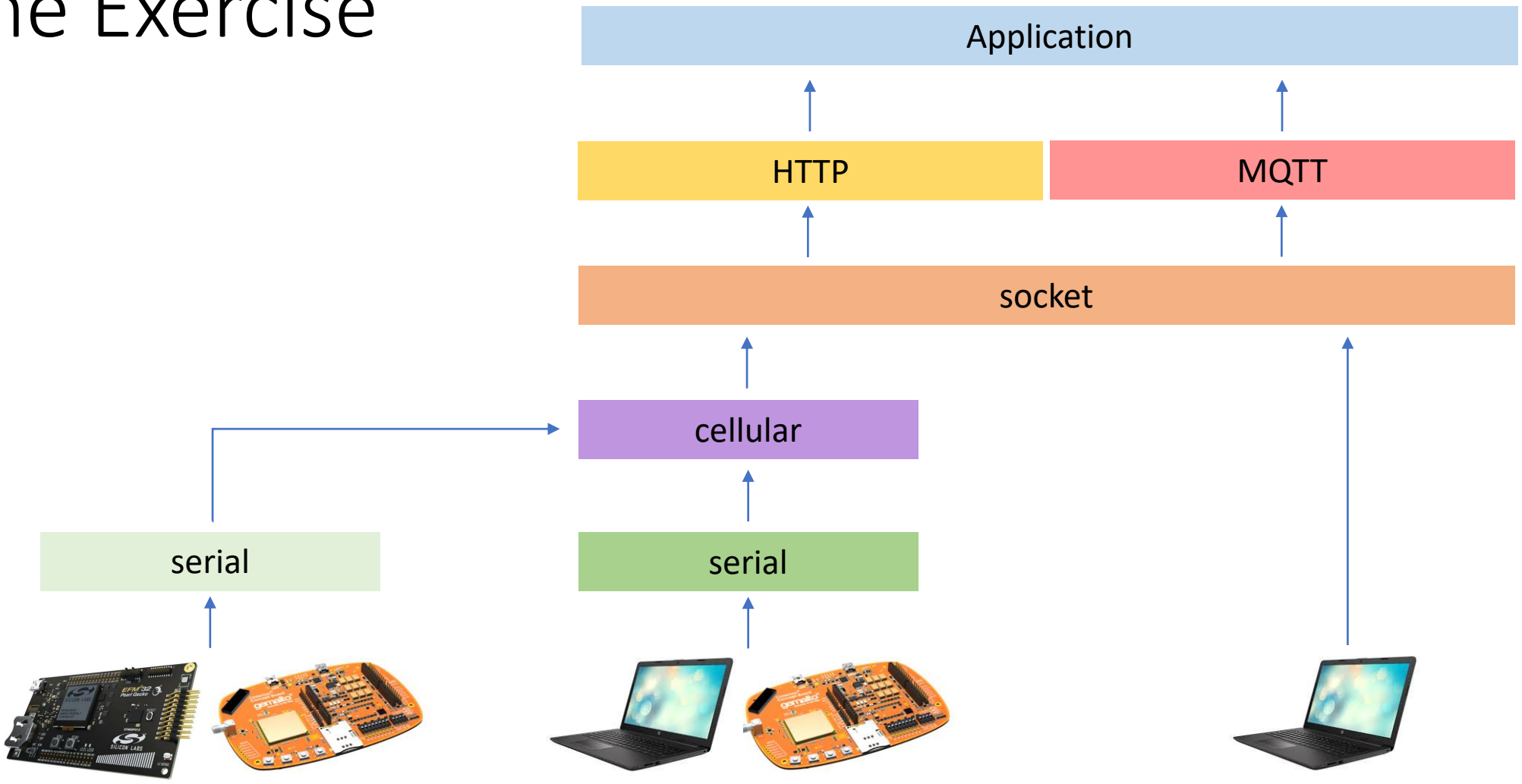
# WORKSHOP ON **INTERNET OF THINGS** 67612

# Exercise 7

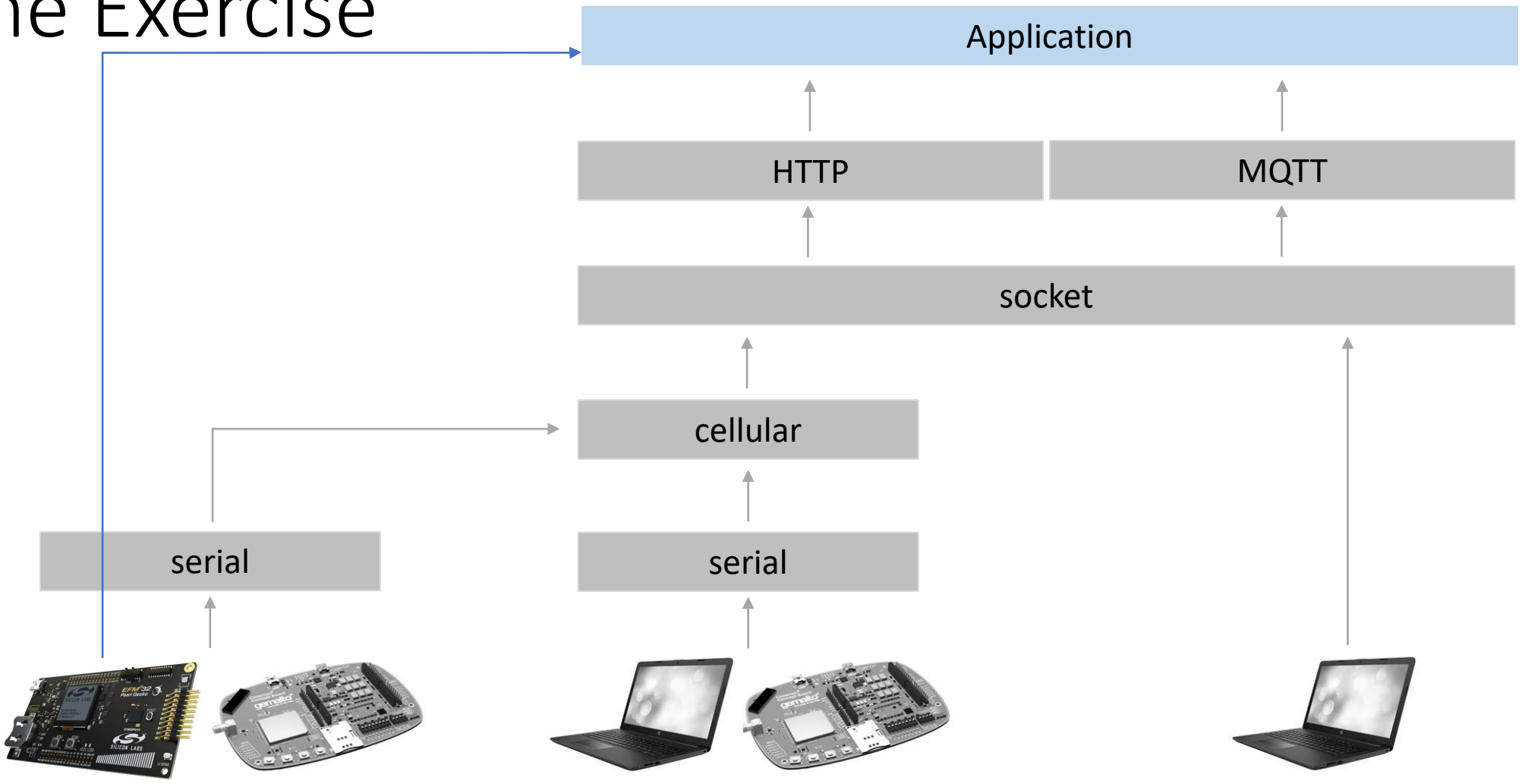
*Hello Embedded World*

Prof. David Hay, Dr. Yair Poleg, Mr. Samyon Ristov

# The Exercise



# The Exercise



# The Exercise

- Print high-frequency clock on the LCD
- Blink LEDs
- React to BNT0 and BTN1
- React to timeout

# Guidance

- **Read** [at least] the following chapters in the Pearl Gecko's Reference Manual:
- **Chapter 2 - System Overview**.
- **Chapter 10 - CMU - Clock Management Unit**. In short: every peripheral we use (e.g., LEUART, USART, GPIO etc.) depends on a clock.
  Part of the magic that makes this MCU extremely low-power is that we can enable/disable the clock for almost every peripheral in the system independently. Not using the LEUART? Shut it down! etc. The system has two main clock domains, the High-Frequency domain (HF) and the Low-Frequency Domain. Each of them can be generated either by an external crystal oscillator ("XO") or an internal Resistor-Capacitor Oscillator ("RCO"). So, we have HFXO, HFRCO and then LFXO, LFRCO etc.
  How do you know which oscillators needs to be activated? read carefully the "Functional Description" section and zoom in on Figure 10.1.
- **General tip**: use a proper PDF reader (e.g., Adobe Acrobat Reader), and not a browser. Then you can navigate through the chapters/sections easily.

# Guidance cont’d

- Write code that does the following:
  - Run an infinite loop
  - Print the menu screen which explains what BTN0 and BTN1 do
  - Clicking on BTN1 will cause LED0 and LED1 to start blinking interchangeably (LED0 on & LED1 off, LED0 off & LED1 on. Changing state every ~500ms) and will print on the LCD screen the number of blinks starting from the last click of BTN1
    - Clicking on BTN1 again will start the blinking and counting process from scratch
  - Clicking on BTN0 will stop the LEDs from blinking and will print the HFXO’s frequency on the screen
  - If nothing is clicked for 10 seconds, print again the menu (what BTN0 and BTN1 do), instead of the previous display

  - Use interrupts to handle button-pressed events
  - Use timer interrupt to count the seconds
  - Make the interrupt handlers as short as possible, without any heavy actions
    - For example, printf is heavy and shouldn’t be used in an interrupt
    - GPIO isn’t heavy and can be used in interrupts
    - Changing global variables, that will be handled in the main loop, is OK in interrupts

# Guidance cont’d

- Use Simplicity Studio v5. Note that it generates code that is very different from Simplicity Studio v4
- The following example-projects can help you:
  - Platform - Blink Bare-metal
  - Platform - Simple Button Bare-metal
  - Platform - Sleeptimer Bare-metal
  - Platform - MEMLCD Bare-metal

# Guidance cont'd

- Suggestion: Implement your own printf
  - You use the existing sdtio.h printf, but that requires some modifications that aren't that easy to do (redirecting the output to the LCD screen)
  - If you install Simplicity Studio v4, many example projects print on the LCD screen and already did the required work and redirected stdio.h printf to LCD (e.g., textdisplay). You can try and copy their code
  - You can use 3rd party software that implement printf, e.g.:
    - SEGGER RTT printf
    - Legacy Printf
    - Tiny printf
  - Note that 3rd party software still requires redirecting the output to the screen
  - The "Platform - MEMLCD Bare-metal" example project shows how to print on the screen. From there, creating your own printf ("myprintf" or some other name) is straightforward

# Exercise #7

- Work & submit in pairs
- Deliverables:
  - Provide .sls and .bin files (project source code and definitions, and the compiled version)
  - A README file with your names, email addresses, IDs and adequate level of documentation of the deliverables and software design-architecture-flow description
  - If anything special is needed (compilation instructions and environment requirements), add it to the README
- Pack all the deliverables as .zip or .tar and upload to Moodle
- Deadline: 14.12.20, 23:59
- The grade will be based on code's functionality, description, and clear implementation

# Contact

- Moodle's 'Workshop Discussions' forum is the best place for questions.

- But if needed, contact us personally:
- David Hay – dhay@cs.huji.ac.il
- Yair Poleg – yair.poleg@mail.huji.ac.il
- Samyon Ristov – samyon.ristov@mail.huji.ac.il