UNIT 21: Hardware & Software Integration

Estimated Time in Hours: 10

Big Idea(s)	Enduring Understandings	Projects & Major Assignments
5 System Security 2 Establishing Trust	5.1	- Demonstrate knowledge of both software and hardware by writing a simple python program to blink an LED on a computer-connected breadboard. - Research how internal and external hardware interacts with software. - Research types of malware and hardware vulnerabilities.

Guiding Questions:

- Does hardware depend on software for computing?
- Without hardware, how does software work?
- How does internal and external hardware interact with software?
- What are the categories of malware?
- Does hardware matter for malware?
- What is the difference between low level and high level programming languages?
- What does a software engineer need to consider when developing software for a specific hardware platform?
- How does domain separation relate to hardware and software?

Learning Objectives & Respective Essential Knowledge Statements	Materials	Instructional Activities and Classroom Assessments
5.1 EU: Systems consist of a combination of hardware and software that together achieve some objective and security requires integration of both. 5.1.1 LO: Students will identify how hardware and software	• Computer, lecture slides, projector, graphic organizers, access to Internet • How hardware and software work together: "How Computers Work: Hardware and Software."	• Show the YouTube video and use a video viewing guide to assess learning. • Ask students how hardware and software work together.

work together in complex ways to achieve an overall objective.	*YouTube*, uploaded by Code.org, 30 Jan 2018, https://youtu.be/xnyFYiK2rSY	
5.1.1b EK: Neither hardware or software is useful without the other. 5.1.1e EK: Software includes programs written to run on servers, laptops, and traditional computers. Computing devices accomplish no tasks without running software that tells it what to do.	• Examples of hardware and software and how they require each other: "Computer Science Basics: Hardware and Software." *YouTube*, uploaded by GCFLearnFree.org, 3 Oct 2018, https://youtu.be/vG_qmtdBPTU	• Explain that a computer must consist of both hardware and software. • Ask students to explain what their phone could do without any software. • Show the linked YouTube video to emphasize that without software, the computer is just a pile of electronics. Without hardware, the software instructions will not achieve any results. • Task students with providing concrete examples of software and hardware working together.
5.1.1c EK: Software instructions may manipulate data, manipulate physical systems or manipulate both. For example, software in a vehicle may record the vehicle speed and send it to a cloud storage system, other software may cause the brakes to be physically applied and reduce the speed, and still other	• How software interacts with hardware, and vice-versa: Bair, Bettina. "Inside your computer." *YouTube*, uploaded by TED-Ed, 1 July 2013, https://youtu.be/AkFi90lZmXA	• Explain the details of how software interacts with hardware. • Show the linked YouTube video and review how the instructions from a mouse interacts with software and hardware again. Use a video viewing guide if necessary. • Optionally, use hardware/build a PC graphic organizers from Unit 3 here. Task students with classifying how internal and external hardware interact with software. This can be a research project.

software may both record and manipulate the vehicle speed.		
5.1.1d EK: Malware, short for malicious software, is any software intentionally designed to cause damage to a computer, server, client, or computer network.	• Malware overview and categories: "Malware: Difference Between Computer Viruses, Worms and Trojans." *YouTube*, uploaded by Kaspersky, 21 Mar 2016, https://youtu.be/n8mbzU0X2nQ	• Revisit malware and dive into the different categories: virus, worm, botnet, etc. • Ask students how malware impacts hardware. • Assign students a type of malware to research. They should summarize how it infects, its goals, and some examples or implementation of the malware.
5.1.1f EK: Software can be written in high level languages such as Python, C, Perl, Java and the high level software is converted into low level instructions that tell the CPU, memory, and other devices exactly what to do.	• Example software-hardware integration program: "Making a LED blink using the Raspberry Pi and Python." *RaspberryPiHQ.com*, 11 Jan 2018, https://raspberrypihq.com/making-a-led-blink-using-the-raspberry-pi-and-python/	• Describe high level programming languages. • Compare and contrast popular languages. • Challenge students by having them write a simple program in Python to blink an LED wired to a computer-connected breadboard. This lab will demonstrate how software can interact with hardware.
5.1.1g EK: Software can be written in low level machine specific instructions that tell the CPU, memory, and other devices		• Explain low level languages and compare and contrast with high level languages. Point out their advantages and disadvantages.

exactly what to do (e.g. add memory locations one and two and store the result in memory location.		• Show the students examples of each. Ask them which language they would prefer to use to develop software. Why?
5.1.1i EK: Embedded software is computer software, written to control machines or devices that are not typically thought of as computers, commonly known as embedded systems. 5.1.1h EK: Embedded software can be built directly into the physical device so the instructions on how a device will behave are physically part of the device and often cannot be changed without changing the hardware itself.		• Introduce the concept of embedded systems alongside examples and use cases. • Explain how sometimes the hardware and software are tightly integrated in embedded software. • Provide examples of embedded software. Explain how this is similar to an operating system and provides unique optimizations. • Ask students to explain any drawbacks of embedded software.
5.1.1j EK: Software ultimately relies on the physical hardware to accomplish its task and even if the software is written perfectly, it will not perform the desired function if the hardware fails to behave as expected. In other words, the software may correctly instruct the hardware to add two numbers and store the result in memory location 3. If memory location 3 has an error		• Revisit how hardware and software depend on each other to properly compute. • Ask students: in the case of a desktop computer, what happens if the hard-drive fails? What happens if the power supply fails? What happens if the fans fail?

or vulnerability and does not store the correct value, the software will not accomplish its objective.		
5.1.1k EK: Hardware ultimately relies on the software instructions to accomplish its task and even if the hardware operates perfectly, it will not perform the desired function if the software fails directs it to execute the wrong instructions. In other words, the hardware may be able to correctly apply the brakes in a vehicle when instructed to do but it will not prevent a vehicle crash if the software is too slow in deciding when to apply the brakes.		• Conversely, what happens if software fails? What happens your web browser crashes? What happens your operating system fails? • Ask students if they think hardware designers need to be aware of software. Do software designers need to be aware of hardware?
5.1.1l EK: The overall system can be manipulated to act incorrectly if there is a vulnerability in the hardware, the software, the interface between them, or any combination of those.	• Review of Spectre and Meltdown: "Meltdown & Spectre vulnerabilities – Simply Explained." *YouTube*, uploaded by Simply Explained, 15 Jan 2018, https://youtu.be/bs0xswK0eZk	• Review hardware vulnerabilities like Spectre and Meltdown. • Ask students if hardware vulnerabilities are easier or harder to detect than software vulnerabilities. • Have students research specific hardware vulnerabilities. What does it do? How long was the vulnerability in existence before it was discovered? Is there a patch for the vulnerability, and what does it require?

2.3.1 LO: Students will give examples of the principle of domain separation, which allows for the enforcement of rules governing the entry and use of domains by entities outside the domain. 2.3.1a EK: A domain refers to a collection of data or instructions that warrant protection. 2.3.1b EK: Communications between domains are allowed only as authorized.		• Review the principle of domain separation. • Ask students to explain how domain separation deals with software and hardware. • Are software and hardware domains? When should they remain separate, and how should they be used in tandem?
8.1.1h EK: Cybersecurity events have led to the development of various cybersecurity career paths and various needs in order to prepare people for these new types of jobs.		• Explore a relevant career, such as information systems security developer.