BitWands: Interactive CS Learning System

Evan Morse
Senior Projects
12.3.25

Project Proposal

Table of Contents

---

1. Project Summary

BitWands is an educational tool consisting of eight wireless handheld devices that function as an interactive binary representation system. The system includes a central hub and accompanying game software. Each wand represents a single bit—students raise or lower their wands to collectively represent hexadecimal digits. The physical interaction helps students understand binary numbers, hexadecimal notation, and bit manipulation.

Many students struggle with abstract binary concepts taught through traditional paper exercises and diagrams. BitWands provides a hands-on alternative where students physically embody the

data representation, making the learning process more concrete and collaborative.

The project includes detailed documentation of the design process, covering bare-metal embedded systems development, custom HAL implementation, linker scripts, and build systems. All hardware designs, software, and documentation will be released under open-source licenses. This allows the product to not only be an education tool by design, but also an education tool in its design with all documentation serving as a teaching document.

---

2. Problem Statement

2.1 Educational Gap

Binary representation, hexadecimal notation, and bitwise operations are abstract concepts. Students typically learn them through paper exercises, diagrams, and explanations. Many students, particularly those who learn better through physical activity, have difficulty internalizing these concepts.

2.2 Validation Through Testing

In summer 2025, I tested this concept with middle school students at a science camp using physical flags to represent bits. Students played games involving binary representation and basic circuit modeling. They showed good engagement and seemed to understand the concepts more quickly than I expected. This suggested that physical, collaborative representation might be effective for teaching these topics. This was in stark contrast to the low engagement levels of previous years using less interactive, more traditional methods.

2.3 Lack of Available Tools

I have found few existing products that provide this type of collaborative, physical learning experience, and nothing for teaching CS subjects. Most CS educational technology consists of software simulations or traditional materials. Physical computing tools tend to focus on programming robots or single-user interfaces rather than multi-user collaborative systems. Considering computers are responsible for creating the video game industry, it almost feels negligent that computers have not been appropriately leveraged to gamify the learning of more computer science subjects.

---

3. Project Objectives

1. **Design and build eight functional electronic wand devices** with orientation sensing, visual feedback (LED indicators), and wireless communication capabilities
2. **Develop a central hub system** to coordinate all wands and interface with game software
3. **Create game software** supporting multiple game modes including competitive team-based and cooperative whole-group configurations
4. **Write comprehensive educational documentation** covering bare-metal embedded development, custom HAL design, linker scripts, and build systems
5. **Demonstrate integration of analog sensing, digital control, embedded programming, wireless communication, and user interface design**
6. **Release all hardware designs, software, and documentation** under open-source licenses to benefit the broader educational community

---

4. Innovation and Unique Features

4.1 Key Features

- **Physical representation:** Students use their bodies and physical devices to represent abstract binary concepts
- **Collaborative learning:** Eight students work together as a byte, requiring coordination and communication
- **Dual purpose:** The hardware teaches CS concepts while the design process itself serves as an embedded systems case study
- **Complete documentation:** All design work from bare metal up will be documented as a learning resource
- **Open source:** All designs, code, and documentation will be freely available

4.2 Inspiration and Methodology

The project is inspired by Charles Petzold's "Code," which builds computing concepts from simple fundamentals. The collaborative mechanics draw from "The Three-Body Problem," where multiple entities must coordinate to convey information in a binary system of flags. The combination provides a different approach to teaching these concepts.

---

5. Technical Design

5.1 System Architecture

The BitWands system has three main components: the wand devices, a central hub, and game software. Each wand communicates with the hub via radio. The hub connects to a computer via USB and sends wand states to the game software, which handles the visual interface, scoring, and game logic.

5.2 Wand Hardware Design

Core Components

Component	Specification
Microcontroller	Raspberry Pi RP2040 (dual-core ARM Cortex-M0+, 133 MHz, 264KB RAM)
Radio Module	RFM69HCW (433/868/915 MHz, packet-based, up to 300m range)
Orientation Sensor	Tilt/orientation sensor (accelerometer or custom tilt switch circuit)
Visual Feedback	RGB LEDs (green for raised/1 state, red for lowered/0 state)
Power Supply	Rechargeable battery (LiPo or similar) with charging circuit

Orientation Detection

The wand must reliably detect when it is raised (vertical, representing binary 1) versus lowered (horizontal or downward, representing binary 0). Two approaches are under consideration:

- **Accelerometer/IMU Module:** Provides precise orientation data but adds cost and complexity
- **Custom Tilt Switch Circuit:** Simple, low-cost solution using mercury-free tilt switches or ball-bearing mechanisms

A bonus objective is to design and implement a custom tilt detection circuit without using pre-built sensor modules, demonstrating analog circuit design skills, as time allows.

5.3 Hub Hardware

The central hub uses a similar RP2040 microcontroller and RFM69 radio module. It receives packets from all eight wands, aggregates their states, and communicates with the game software via USB serial connection. The hub maintains synchronization, handles packet loss, and ensures low-latency updates to provide responsive gameplay. A secondary design option being considered is the use of single-board computer like the Raspberry-Pi to interface with an RFM69 module and act as the hub. This second option may be faster to implement, and would mean only one board has to be designed.

5.4 Software Architecture

Embedded Firmware

The wand and hub firmware will be developed from first principles, starting with bare-metal programming on the RP2040. This approach serves the dual purpose of creating functional firmware while providing comprehensive educational content on embedded development. Key components include:

- Custom linker scripts defining memory layout and startup code
- Hardware Abstraction Layer (HAL) written from scratch for GPIO, SPI, timers, and interrupts
- Radio communication protocol with packet framing, addressing, and basic error detection
- State machine managing sensor reading, LED control, and radio transmission
- Custom build system using Make or CMake to compile and link the firmware

Game Software

The initial game software will be a simplified version that runs in a Linux terminal, and provides the user interface, game logic, and educational content. Multiple game modes will be supported:

- **4v4 Competitive Mode:** Two teams of four compete to match single hexadecimal digits (0-F) as quickly as possible
- **8-Bit Cooperative Mode:** All eight players work together to represent full bytes (two hex digits, 00-FF) in sequence
- **Training Mode (if time allows):** Display current binary and hex values, allowing free practice without time pressure
- **Challenge Mode (if time allows):** Progressive difficulty levels teaching binary-to-hex conversion, bit manipulation, and bitwise operations

---

6. Educational Documentation

The project will include detailed documentation of the design process. This serves two purposes: it forces me to understand everything thoroughly, and it provides a resource for others learning embedded systems. The documentation will cover:

- Setting up a bare-metal ARM Cortex-M development environment
- Writing linker scripts (memory layout, sections, startup code)
- Building a Hardware Abstraction Layer from scratch (register access, peripheral initialization, driver structure)

- Build systems (Make/CMake configuration, toolchain setup)
- SPI communication implementation at the register level
- Radio communication protocols (packet structure, error handling, synchronization)
- PCB design for mixed-signal systems (power distribution, signal integrity, layout)

This documentation turns the project into a learning resource rather than just a finished product.

---

7. Design Integration Across Disciplines

As a Computer Engineering student, this project needs to cover both electrical and software design. The split works out to roughly 50-50.

7.1 Electrical Engineering Components (≥50% of design)

- **Analog:** Orientation sensor design (if doing custom circuit), power regulation, battery charging
- **Digital:** Microcontroller peripheral setup, SPI bus, GPIO control
- **Communications:** Radio module integration, packet radio protocol, antenna considerations
- **Power:** Battery selection, power consumption analysis, LED driving
- **PCB Design:** Schematic and layout for wand and hub boards

7.2 Software Engineering Components

- **Embedded Programming:** Bare-metal firmware, custom HAL, interrupt handlers
- **Application Software:** Game interface, serial communication, game logic
- **Build Systems:** Linker scripts, Makefiles, toolchain configuration
- **Protocol Design:** Packet format, state synchronization, error handling

This covers Analog, Digital, Microcontrollers, Power, and Communications—hitting the required disciplines.

---

8. Testing and Evaluation

8.1 Hardware Testing

Test	Method	Success Criteria
Orientation Detection	Test wand at various angles with oscilloscope/logic analyzer	Correct state detected within 100ms, no false triggers

Test	Method	Success Criteria
Radio Range	Measure packet reception rate at increasing distances	>95% packet delivery within 10m, >80% within 20m
Battery Life	Continuous operation test with current monitoring	Minimum 2 hours of active gameplay per charge
LED Visibility	Visual inspection in various lighting conditions	Color clearly distinguishable from 5m in classroom lighting
Multi-Wand Sync	Operate all 8 wands simultaneously, log timing data	All wand states updated on hub within 200ms, no lost data

8.2 Software Testing

- **Unit Testing:** Individual firmware modules tested in isolation (GPIO, SPI, radio driver)
- **Integration Testing:** Wand-to-hub communication tested with protocol analyzers
- **Game Logic Testing:** Automated tests for scoring, state transitions, and edge cases
- **Latency Testing:** Measure time from wand motion to on-screen update (target <300ms)

8.3 User Testing

I'll test the final system with actual users (students or volunteers) to see if it actually works as a teaching tool. I'll measure basic metrics like whether they understand the concepts, whether they're engaged, and whether they learn anything (simple pre/post test). Also collect feedback on what works and what doesn't.

---

9. Bill of Materials and Budget

9.1 Per-Wand Components (×8)

Component	Part Number	Unit Cost	Total
RP2040 Module	Pico/KB2040	$4.00	$32.00
RFM69HCW Radio	RFM69HCW-433	$6.00	$48.00
Accelerometer/Tilt Sensor	ADXL345 or custom	$3.00	$24.00
RGB LED	WS2812B	$0.50	$4.00
Battery + Charger	LiPo 500mAh	$5.00	$40.00
PCB, passives, misc	Various	$5.00	$40.00
Enclosure/Wand Body	3D printed/PVC	$3.00	$24.00

Component	Part Number	Unit Cost	Total
Wand Subtotal (×8)			**$212.00**

9.2 Hub and Miscellaneous

Item	Unit Cost	Total
Hub (RP2040 + RFM69 + USB)	$15.00	$15.00
Development tools, prototyping supplies	$30.00	$30.00
Contingency (15%)	-	$39.00
Total Project Budget		**$296.00**

The estimated project cost is about **$300**. This covers eight wands, a hub, and prototyping supplies. Component prices are based on typical costs from Adafruit, SparkFun, and Digi-Key.

---

10. Project Timeline and Milestones

10.1 Phase 1: Research and Design (Weeks 1-3)

- Complete literature review and component research
- Finalize orientation sensing approach (accelerometer vs. custom circuit)
- Complete schematic design for wand and hub
- Define communication protocol and packet structure
- Order initial components for prototyping

10.2 Phase 2: Bare-Metal Firmware Development (Weeks 4-7)

- Set up development environment and toolchain
- Write linker script and startup code
- Implement custom HAL for GPIO, SPI, timers
- Develop RFM69 radio driver
- Test basic radio communication between two modules
- Document HAL and build system as educational content

10.3 Phase 3: Hardware Prototyping (Weeks 6-9)

- Build breadboard prototype of single wand
- Test orientation sensing and LED control

- Design and order PCBs for wand and hub
- Assemble and test first PCB revision
- Design and prototype physical wand enclosure

10.4 Phase 4: System Integration (Weeks 10-12)

- Implement complete wand firmware with state machine
- Implement hub firmware with multi-wand coordination
- Test all eight wands communicating simultaneously
- Develop basic game software and USB serial interface
- Conduct initial system integration testing

10.5 Phase 5: Software and Game Development (Weeks 11-14)

- Implement game modes (4v4 competitive, 8-bit cooperative)
- Develop game UI with score tracking and visual feedback
- Create training and challenge modes
- Polish user experience and add sound effects

10.6 Phase 6: Testing and Documentation (Weeks 13-15)

- Complete all hardware and software testing per test plan
- Conduct user testing with volunteer students
- Write final educational documentation
- Create project website with design files and documentation
- Prepare presentation materials and demo

10.7 Phase 7: Final Presentation (Week 16)

- Final project presentation and demonstration
- Submit all deliverables and documentation
- Release complete project under open-source licenses

This timeline assumes about 16 weeks from approval to final presentation. Some phases overlap to save time, and there's buffer time for when things inevitably go wrong. Some preliminary work has already been completed (bare-metal build and source code) in the pre-project exploration phase.

---

11. Risk Analysis and Mitigation

Risk	Impact	Mitigation Strategy
Radio interference or range issues	High - could prevent multi-wand operation	Early testing of radio module, implement robust error handling, consider frequency hopping
Orientation sensing unreliable	Medium - degrades user experience	Test multiple sensor options early, implement software filtering and calibration
PCB design errors	Medium - requires redesign and delays	Thorough breadboard prototyping first, peer review of PCB design, order small batch initially
Component supply chain delays	Medium - could delay project timeline	Order components early, identify alternative parts, maintain component inventory buffer
Battery life insufficient	Low - inconvenient but not critical	Implement power-saving features, use efficient LED driving, select larger battery if needed
Firmware bugs in bare-metal code	Medium - difficult to debug	Incremental development with testing at each step, use debugger and logic analyzer, maintain clean modular code

---

12. Future Development and Commercialization

12.1 Long-Term Vision

This project is a first step toward building better CS education tools. The goal is to create teaching materials that make abstract concepts more concrete. BitWands is a test of whether this physical-digital approach actually works.

12.2 Potential Extensions

- **Additional game modes:** Boolean logic, shift/rotate operations, bitwise arithmetic, or binary logic circuits.
- **Curriculum materials:** Lesson plans, teacher guides
- **Advanced topics:** Digital logic basics, state machines, simple CPU operations
- **Mobile app:** Run the game on tablets/phones instead of computers for better teaching accessibility.
- **Scalability:** Support 16-bit or 32-bit with more wands

12.3 Educational Company Vision

Long-term, I'd like to start a company that makes human focused CS education tools and teaching frameworks. The idea is to create quality teaching materials and release them as open source while making money through support, training, and packaged solutions. Whether this is realistic remains to be seen.

---

13. Open Source Commitment

When the project is done, I'll release everything under open-source licenses:

- **Hardware:** Schematics, PCB layouts, enclosure designs (CERN OHL or similar)
- **Firmware:** All embedded code, HAL, build system (MIT or Apache 2.0)
- **Game software:** Complete source code (MIT or GPL)
- **Documentation:** All design notes and tutorials (Creative Commons BY-SA)

This makes the project useful to others and lets people improve on it. It also fits with my goal of improving CS education broadly and having a positive impact on the world of free information which has served me (and most of us) greatly.

---

14. Conclusion

BitWands is a system for teaching binary and hexadecimal concepts through physical, collaborative interaction. It addresses a real gap in how these topics are taught, based on my testing with middle school students. The project combines building a functional teaching tool with documenting the entire embedded systems design process.

The project covers multiple engineering disciplines: analog circuits, digital systems, microcontroller programming, wireless communication, and power management. The bare-metal firmware approach and documentation show technical depth while creating a useful learning resource.

The preliminary testing worked well, the technical approach is solid, and the project plan is realistic. The budget is about $300, the timeline is 16 weeks, and I've identified the main risks. The project meets the course requirements for design work across multiple disciplines.

By releasing everything as open source, the project can be useful beyond just my senior design work. It might help improve how these concepts are taught, or at least provide a working example of this approach.