

ECE 215/S – Principles of Electronic Cyber Warfare

Block 2: Signal Processing

Lesson 26: Project 2 Workday 1

Department of Electrical and Computer Engineering
United States Air Force Academy

Fall 2026

Where We Left Off

Block 2's new material is done. You now have both halves of what today needs:

- **Lesson 22** — a distance sensor you **calibrated yourself**, good to ± 0.5 cm, living in a function called `distanceCm()`
- **Objectives 2.7–2.8** — gates and truth tables, then FSMs: states, transitions, Moore vs. Mealy, and the seven-step design process

Today your sensor stops reporting range and starts making decisions.

The Mission

You are building the **engagement logic of a short-range fire-control system** — out of the parts already on your bench.

On the bench	In the scenario
Sharp IR sensor on `A0`	the **seeker's range channel**
Serial Monitor	the **scope display** — one line per tick
Your keyboard	the **operator console** — lock and pickle
Jumper, pin 2 to `GND`	the **master arm plug**
`DAC1`, read by a DMM	**telemetry to the weapons panel**
`LED_BUILTIN`	the **launch indicator**

Lesson 24's *safe, searching, locked, engaged* machine — arming here is an *input*, not a state.

Educational exercise — not an actual weapon system or its release logic.

Start With the Demo — It Is the Spec

usafa-ece.org/apps/fsm-demo/ $\rightarrow$ tab **“Mealy · Fire Control Radar”**

- **This is the machine you are about to build in hardware:** same states and encodings, same five inputs, same priority order, both loss edges. Telemetry volts sit in the bubbles, and **FIRE** $=$ **LOCK** $\cdot P \cdot E \cdot M$ **goes on an arrow**
- Out of the box it **steps the spec**: set D, E, M, press L or P, step. It prints the same `MISSILE AWAY` and `NO FIRE` lines, and it **reacquires after a shot**
- The arcs, their labels, the reference table and **Show Answer** sit behind an **Instructor reveal** that we run in class
- On your own, use it to **test** your logic: predict from your own table, step, compare. **Test All Rows** checks your equations against the machine

The demo checks your logic. The diagram and the table are still yours.

States, Encoding, and the Telemetry Output

State	Q_1Q_0	The seeker is…	DAC1
SEARCH	00	scanning; nothing inside the gate	0.8 V
DETECT	01	tracking a target, not locked	1.6 V
LOCK	10	holding the target	2.4 V
(illegal)	11	never entered by design	—

- Three states in **two bits** — the same situation as Lesson 25's worked example, with the same leftover code
- The weapons panel reads that voltage to know what the seeker is doing

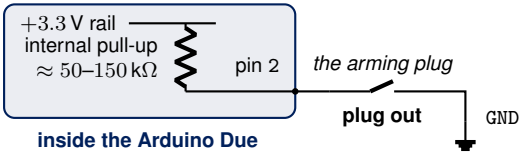
The code 11 exists in the hardware whether your design uses it or not — **leave the starter's guard in.**

Five Inputs, Read Once Every 2 s Tick

	Name	From	Is 1 when
D	target detected	`distanceCm()`	$d \leq 12.0$ cm — **detection gate**
E	in envelope	the same distance	$d \leq 6.0$ cm — **engagement envelope**
M	master arm	pin 2, `INPUT_PULLUP`	the plug is **inserted** (pin reads LOW)
L	lock command	`'l'` or `'L'` in Serial Monitor	the key arrived since the last tick
P	pickle	`'p'` or `'P'` in Serial Monitor	the key arrived since the last tick

- **The envelope sits inside the gate.** E true forces D true. A status line reading `D=0 E=1` means your thresholds are backward
- **L and P are one-shots.** Keys pressed between ticks wait in the serial buffer, set their flag once, and are cleared at the end of that tick. **Holding a key does not hold the input high**

Master Arm: Why INPUT_PULLUP Makes SAFE the Default



plug IN → pin pulled to GND → LOW → **ARMED**
plug OUT → the internal pull-up wins → HIGH → **SAFE**

- An **empty socket reads HIGH**, and we define HIGH as SAFE
- A pulled plug, a broken wire, or a pin you forgot to configure **all read SAFE**
- Wire it the other way — +3.3 V for ARMED — and a **disconnected wire arms the weapon**

Fail-safe is a wiring decision. Nothing real is wired the other way.

The Transitions You Must Write

Order inside each state is part of the spec. Write them as `if` / `else if` in this order.

From SEARCH

1. $D \rightarrow$ **DETECT**
2. else stay

A lock command with no target is not a transition.

From DETECT

1. $\overline{D} \rightarrow$ **SEARCH** *(lost)*
2. else $L \rightarrow$ **LOCK**
3. else stay

Loss beats lock: you cannot lock onto what you cannot see.

From LOCK

1. $\overline{D} \rightarrow$ **SEARCH** *(break lock)*
2. else $P \cdot E \cdot M \rightarrow$ **FIRE**, then **SEARCH**
3. else $L \rightarrow$ **DETECT** *(operator break)*
4. else stay

A pickle that fails an interlock is **not a transition** — the machine stays in LOCK and says why.

In the **table**, rows are tried top to bottom and the **first match wins**, exactly like your `if` / `else if` chain. The Project 2 page has the row format and a template workbook to fill in.

The diagram and the table are yours to produce — one of the three project points.

Two Outputs — One of Each Kind

Output	Kind	Depends on	Behavior
Telemetry on `DAC1`	?	?	0.8 / 1.6 / 2.4 V — one per state
FIRE	?	?	`*** MISSILE AWAY ***`, LED strobes

- FIRE fires only when **all four** hold at once: in **LOCK**, P, E, and M
- A refused pickle prints why and changes nothing — **envelope first, then master arm**:
  `NO FIRE -- OUTSIDE ENVELOPE` `NO FIRE -- MASTER ARM SAFE`
- **Discussion Question 1** asks you to justify these two classifications from the Objective 2.8 definitions and from your code — fill the table in once you can

One machine, one Moore output, one Mealy output. Exactly the Quiz 7 distinction.

Wiring: Two Deltas from the Lesson 22 Lab

Your sensor wiring does **not** change — white→A0, red→5V, black→GND. Add two things:

1. **The master arm plug** — one jumper, digital pin 2 to any GND pin. Start with it *out*
2. **The telemetry tap** — DMM **red** probe on DAC1, **black** probe on GND, meter set to **DC volts**

You should see the DMM read about 0.55 V the moment the sketch starts, before you write anything.
That is the **DAC's floor**, not zero — and it is Discussion Question 2.

DAC1 is an output. Never connect the sensor lead, the jumper, or $+3.3$ V to it.

The Starter Sketch: Five Sections

`EngagementFSM_Starter.ino` — on the course site (Project 2 page).

Section	Whose	What it holds
0 — CONFIG	provided	pins, the 12.0 and 6.0 cm thresholds, the 2 s tick, the LED strobe, ADC and DAC constants
1 — PLUG IN	**yours**	`CAL_K`, `CAL_B`, and the one line of `distanceCm()`
2 — PLUMBING	provided — **do not modify**	sensor read, serial one-shots, `masterArmed()`, `dacWriteVolts()`, `flashLaunchIndicator()`, `printStatus()`, and the tick skeleton
3 — TRANSITIONS	**yours**	the `switch (state)` next-state logic
4 — OUTPUTS	**yours**	the **Moore** telemetry write and the **Mealy** FIRE

Do Section 1 first — nothing below it works until `distanceCm()` returns cm.

Today's Goal, and the Three Ways Teams Lose the Period

By the end of class: the diagram on paper, the table in the workbook, and **transitions working** in the status line, **both loss edges included**. Outputs are next lesson.

- **Calibration constants never plugged in.** `CAL_K` left at `0.0` means `d= --- cm`, D and E stuck at 0, and the machine sits in SEARCH forever — which looks exactly like broken state logic and is not
- **Fighting the one-shots.** A key sets its flag for *one* tick. Holding `'l'` down does not hold L high, and a key pressed twice between ticks still counts once
- **"The pin is floating, so it is unpredictable."** Not with `INPUT_PULLUP` — an empty socket is a **defined** HIGH, and HIGH is SAFE. That is the whole point

Do the table before the code. A table is far cheaper to debug.

Authorized Resources

- Work in your **group of 2** only — not with cadets outside your group
- You **may** use any ECE 215/S course material from this semester, EI from ECE instructors, the Arduino website, the starter sketch, and internet resources
- Internet resources must be **documented**
- **GenAI is Level 4** on this project — co-create and revise. You are responsible for demonstrating understanding of anything you take from it, and we will ask
- You may **not** use materials from previous semesters

Note this is **looser** than the Lesson 22 lab, which was Level 3. Check the policy level per assignment.

Wrap-Up

Reminders:

- **Quiz 7** (Objectives 2.7–2.8) is due at Lesson 27
- **Project 2** is due at Lesson 29 — **check Gradescope for the exact deadline**
- Groups of 2, GenAI Level 4, document everything

Next lesson:

- **Lesson 27 — Project 2 Workday 2.** Outputs, the DMM check, and the demo script