

Engineering Portfolio

Haya Dakhil



CONCENTRAY

Concentray: PCB

Embedded controller / display board for a desk-mounted focus device with screen timers and task tracking.

[Schematic & PCB design, simulations, component selection & BOM, assembly, bring-up, documentation.](#)

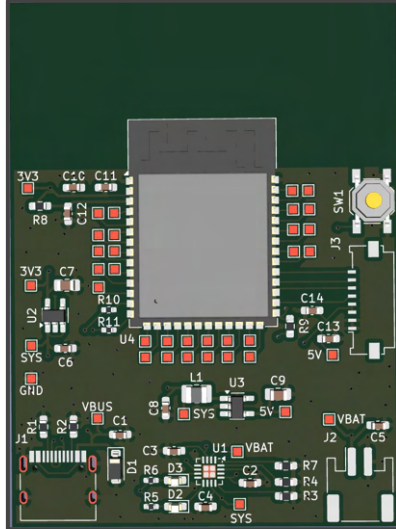
Design includes: 4-layer PCB, ESD protection, DC-DC Buck (3.3V) and Boost (5V) Regulators, SPI communication, Smart power path management, header (V1) / FFC (V2) connectors, SWD, via stitching.

Tools used: KiCad, LTSpice, EE lab tools (oscilloscope, DMM, power supply, LPKF Reflow, SMT/stencil soldering).

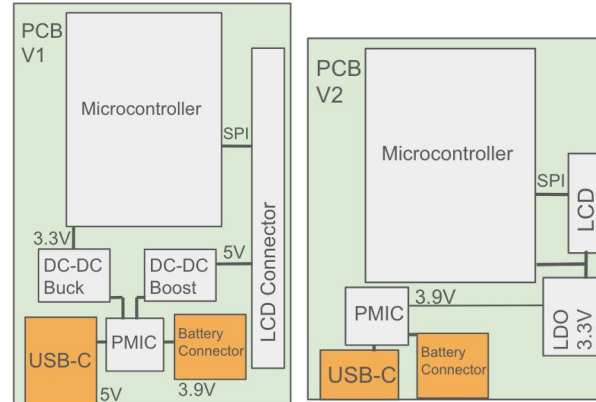
V1



V2



Circuit Block Diagrams





CONCENTRAY

Concentray: Firmware

UI/UX firmware, validated with simulation to bypass the need for microcontroller hardware.

[Wireframing](#), [UI/UX principles](#), [widget drawing](#), [navigation logic](#), [simulation](#), [documentation](#)

Firmware includes: Timer countdown, scrolling navigation, button animations, start/stop/pause/resume.

Software / Frameworks: C++, LVGL graphics library, Arduino, PlatformIO, SDL.

Tuesday, July 15
10:00 AM

You have
4 tasks today

Press button to view tasks

Let's get something done today.

Review document

Review the latest draft and leave concise comments.

02:52

Blocked

Pause Task

Today's Overview
Tuesday, July 12

10:00

Review document

Write report

Write email

Not started

Respond to messages

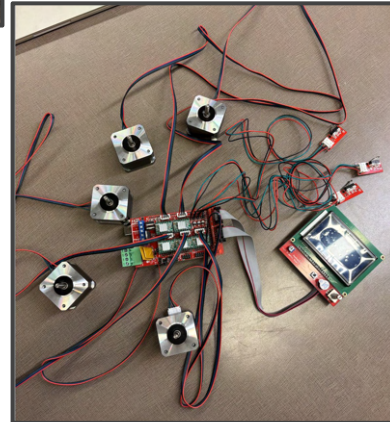
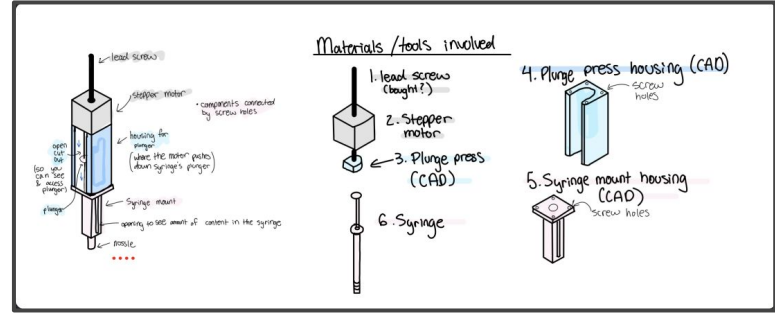
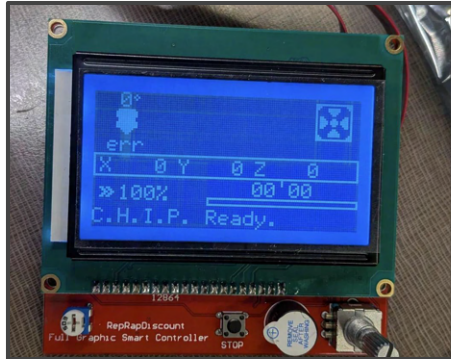
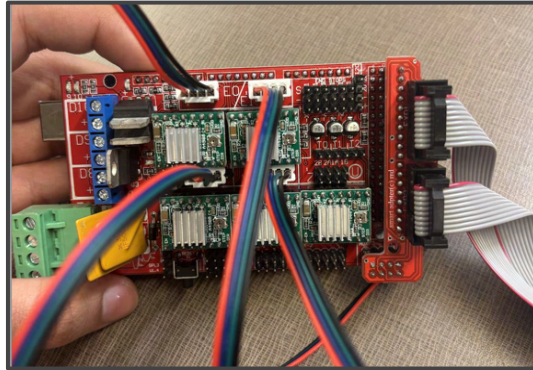
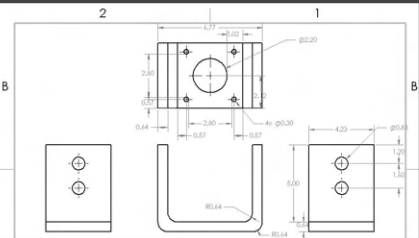
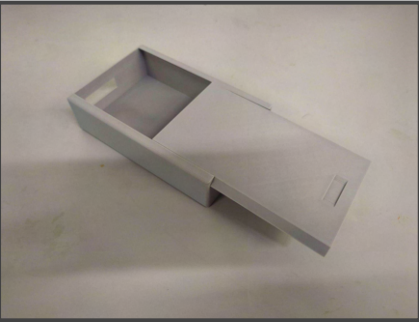
```
g++ main.cpp -o main
22 #ifndef ARDUINO
23 int main() {
117
118     while (SDL_PollEvent(&e)) {
119         if (e.type == SDL_KEYDOWN) {
120             break;
121         }
122         if (e.type == SDL_KEYUP) {
123             break;
124         }
125         case SDLK_UP: //arrow up button (rotary encoder up)
126             if (currentScreen == TASK_LIST) {
127                 taskList.previousTask();
128             } else if (currentScreen == TASK_PREVIEW || currentScreen == TASK_RUNNING || currentScreen == TASK_PAUSED) {
129                 tv_obj_scroll_by(taskScreen.getDescriptionContainer(), 0, 20, LV_ANCH_ON);
130             }
131             break;
132         case SDLK_DOWN: //arrow down button (rotary encoder down)
133             if (currentScreen == TASK_LIST) {
134                 taskList.nextTask();
135             } else if (currentScreen == TASK_PREVIEW || currentScreen == TASK_RUNNING || currentScreen == TASK_PAUSED) {
136                 tv_obj_scroll_by(taskScreen.getDescriptionContainer(), 0, -20, LV_ANCH_ON);
137             }
138             break;
139         case SDLK_RETURN:
140             if (currentScreen == TASK_LIST) {
141                 tv_obj_clean(tv_scr_act());
142                 taskScreen.unTask(taskList.getSelectedTitle(), taskList.getSelectedDescription());
143                 taskScreen.bindTaskList(taskList, taskList.getSelectedIndex());
144                 taskScreen.tv_preview_screen();
145                 currentScreen = TASK_PREVIEW;
146             } else if (currentScreen == TASK_PAUSED) {
147                 // stop without resetting elapsed so preview shows remaining time
148                 taskScreen.stop(false);
149                 currentScreen = TASK_PREVIEW;
150             } else if (currentScreen == TASK_PREVIEW) {
151                 // Return to home screen from preview
152                 tv_obj_clean(tv_scr_act());
153                 taskList.tv_main_cont();
154                 taskList.tv_home_screen();
155                 currentScreen = HOME;
156             }
157             break;
158     }
159 }
```

UW NanoRobotics Group

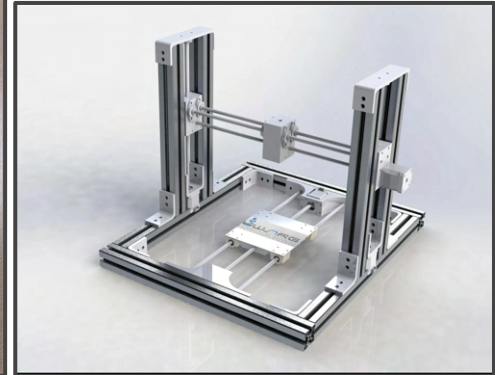
CHIP Project: Developing a conductive hydrodynamic ink printer that uses nanoparticle inks to fabricate PCBs.

PSU Encasing, 5 U-Bracket custom mounts & mechanical drawings, Electronics assembly/validation/measurements, Fluid integration mechanics schematics.

Tools: UW Student Machine Shop, Solidworks, 3D printing, DMM, stepper motor, soldering.

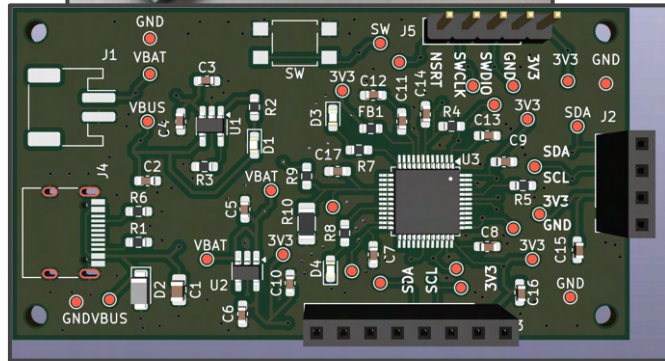
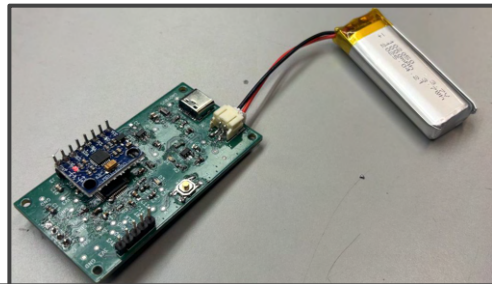


Printer demo



Hardware includes: STM32F103 MCU, MPU-6050 IMU (I²C), SSD1306 OLED (I²C), 3.3 V LDO regulation, Li-Ion charging via USB-C, low-battery sense divider, status LEDs, ESD protection, SWD programming/debugging header.

EE software / tools: KiCad, EE lab tools (oscilloscope, function generator, DMM, power supply, soldering).



Fitness Rep Tracker - Firmware

Wrist-mounted STM32 controller board to detect and classify gym exercises with real-time OLED feedback.

Driver development (I²C, UART, SysTick), real-time filtering, calibration routines, adaptive thresholding, finite-state machines, custom OLED UI.

Firmware includes: IMU (MPU6050) driver with calibration, OLED (SSD1306) display driver, rolling buffer stats, low-pass filters, rep detection with peak/refractory logic, exercise-specific configs, state machine (boot → select → calibrate → run), UART logging, systick timing, and peripheral init with fallback/error handling.

Software / Frameworks: C (PlatformIO), STM32 HAL, FreeRTOS-style FSM (bare-metal), I²C, UART, SDL (sim testing).

Adaptive rep detection & rolling statistics

```
src > sensing > C rep_detect.c
1 //
2 void rep_detect_end_calibration(exercise_t ex, float *out_mu, float *out_sigma)
3 {
4     if (ex == EX_COUNT || calib_count[ex] == 0) return;
5
6     //compute mean and standard deviation
7     float mu = calib_sum[ex] / calib_count[ex];
8     float variance = (calib_sum_sq[ex] / calib_count[ex]) - (mu * mu);
9     float sigma = sqrtf(fmaxf(variance, 0.0f));
10
11     //store in runtime context
12     REP_CTX(ex).baseline_mu = mu;
13     REP_CTX(ex).baseline_sigma = fmaxf(sigma, MIN_SIGMA_FLOOR_G);
14     REP_CTX(ex).calibrated = true;
15
16     //output values
17     if (out_mu) *out_mu = mu;
18     if (out_sigma) *out_sigma = sigma;
19 }
20
21 static void update_rolling_stats(exercise_t ex, float new_sample)
22 {
23     if (ex == EX_COUNT) return;
24
25     //add new sample to buffer
26     sample_buffer[ex][buffer_index[ex]] = new_sample;
27     buffer_index[ex] = (buffer_index[ex] + 1) % ROLLING_BUFFER_SIZE;
28
29     if (buffer_index[ex] == 0)
30     {
31         buffer_filled[ex] = true;
32     }
33
34     //compute mean
35     float sum = 0.0f;
36     uint16_t count = buffer_filled[ex] ? ROLLING_BUFFER_SIZE : buffer_index[ex];
37
38     for (uint16_t i = 0; i < count; i++)
39     {
40         sum += sample_buffer[ex][i];
41     }
42     rep_state[ex].mean = sum / count;
43
44     //compute standard deviation
45 }
```

I²C bus initialization with fallback protection

```
src > drivers > C i2c_bus.c
1 #include "i2c_bus.h"
2 #include "mcu_pinmap.h"
3
4 HAL_StatusTypeDef i2c_bus_init(void)
5 {
6     hi2c1.Instance = I2C1;
7     hi2c1.Init.ClockSpeed = I2C_BUS_SPEED_FAST_MODE;
8     hi2c1.Init.DualMode = I2C_DUALMODE_DISABLE;
9     hi2c1.Init.OwnAddress1 = 0;
10    hi2c1.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
11    hi2c1.Init.DualAddressMode = I2C_DUALADDRESS_DISABLE;
12    hi2c1.Init.OwnAddress2 = 0;
13    hi2c1.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
14    hi2c1.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
15
16    if (HAL_I2C_Init(&hi2c1) != HAL_OK)
17    {
18        //fallback to standard mode if fast mode fails
19        hi2c1.Init.ClockSpeed = I2C_BUS_SPEED_STANDARD_MODE;
20        if (HAL_I2C_Init(&hi2c1) != HAL_OK)
21        {
22            return HAL_ERROR;
23        }
24    }
25    return HAL_OK;
26 }
27
28 HAL_StatusTypeDef i2c_mem_read(uint16_t dev_address, uint16_t reg_address, uint8_t *pdata, uint16_t Size)
29 {
30     return HAL_I2C_Mem_Read(&hi2c1, dev_address, reg_address, I2C_MEMADD_SIZE_8BIT, pdata, Size, HAL_MAX_DELAY);
31 }
32
33 // @brief Writes a sequence of bytes to a device's internal register.
34 // @param i2c_dev_address: I2C device address
35 // @param i2c_reg_address: I2C register address
36 // @param i2c_data: pointer to data to be written
37 // @param i2c_size: size of data to be written
38 HAL_StatusTypeDef i2c_mem_write(uint16_t dev_address, uint16_t reg_address, uint8_t *pdata, uint16_t Size)
39 {
40     return HAL_I2C_Mem_Write(&hi2c1, dev_address, reg_address, I2C_MEMADD_SIZE_8BIT, pdata, Size, HAL_MAX_DELAY);
41 }
42
43 }
```

MPU6050 IMU Initialization & Configuration

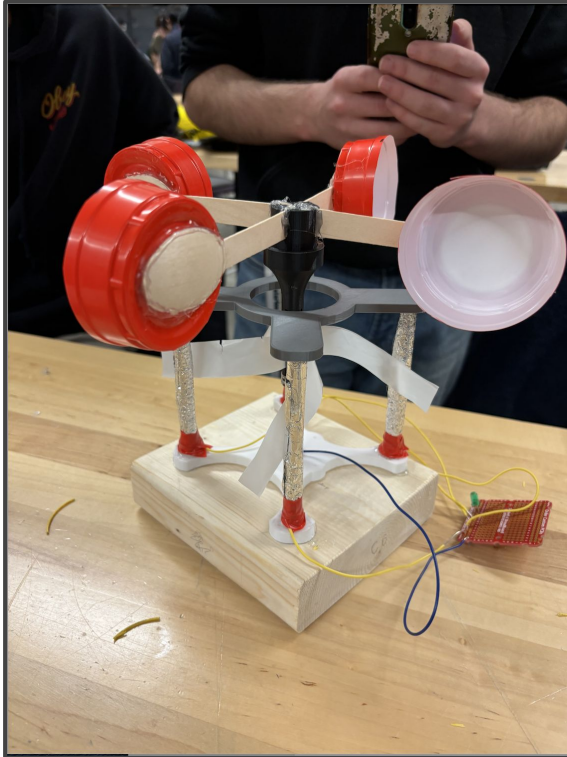
```
28 //initializing calibration data
29 static float accel_bias[3] = {0.0f, 0.0f, 0.0f};
30 static float gyro_bias[3] = {0.0f, 0.0f, 0.0f};
31
32 //writes a single byte to an MPU register.
33 static HAL_StatusTypeDef MPU6050_WriteRegister(uint8_t reg, uint8_t value)
34 {
35     return I2C_mem_write(MPU6050_I2C_ADDR, reg, &value, 1);
36 }
37
38 //reads a single byte from an MPU register.
39 static HAL_StatusTypeDef MPU6050_ReadRegister(uint8_t reg, uint8_t *value)
40 {
41     return I2C_mem_read(MPU6050_I2C_ADDR, reg, value, 1);
42 }
43
44 //initializing MPU
45 HAL_StatusTypeDef MPU6050_Init(void)
46 {
47     uint8_t who_am_i;
48     if (MPU6050_ReadRegister(0x75, &who_am_i) != HAL_OK || who_am_i != 0x68)
49     {
50         // LOG("MPU-6050 not found or WHO_AM_I mismatch!\n");
51         return HAL_ERROR;
52     }
53
54     // Wake up MPU-6050
55     if (MPU6050_WriteRegister(MPU6050_PWR_MGMT_1, 0x00) != HAL_OK) return HAL_ERROR;
56
57     // set sample rate to IMU_SAMPLE_HZ with sample rate being gyroscope output rate / 1 + SMPLRT_DIV
58     // gyro output rate = 3kHz (when DLPF is enabled and set to 42Hz or less)
59     // 200 Hz = 3000 Hz / (1 + SMPLRT_DIV) => 1 + SMPLRT_DIV = 3 => SMPLRT_DIV = 4
60     if (MPU6050_WriteRegister(MPU6050_SMPLRT_DIV, (1000 / IMU_SAMPLE_HZ) - 1) != HAL_OK) return HAL_ERROR;
61
62     // configure DLPF (Digital Low Pass Filter)
63     // 8.4kHz SYNC SET = 0, DLPF_FQ = 3 (42 Hz) for both accel and gyro
64     if (MPU6050_WriteRegister(MPU6050_CFG, 0x03) != HAL_OK) return HAL_ERROR;
65
66     //configure gyroscopes +/- 250 deg/s (PS_SEL = 0)
67     if (MPU6050_WriteRegister(MPU6050_GYRO_CONFIG, 0x00) != HAL_OK) return HAL_ERROR;
68     GYRO_SCALE_FACTOR = 131.0f; // 331 LSB/deg/s for +/- 250 deg/s
69
70     // configure accelerometers +/- 2g (AFS_SEL = 0)
71 }
```

Triboelectric Nano-Generator (TENG)

Won 1st place out of 144 students in the TENG prototyping competition to harvest mechanical energy into electricity.

Solidworks, 3D printing, assembly, materials science, rapid prototyping, cost efficiency

Tools: Soldering, prototyping / building tools (wood block, paper cups, teflon, aluminum, wooden sticks)



[Video Link Here](#)

Bio-Mechatronics Design Team

Developing affordable exoskeleton prosthetics for children with muscular dystrophy.

Base designing & prototyping (ESP32 / servo housing), component / material research, exoskeleton assembly.

Tools: EMG electrodes, Solidworks, 3D printing, servo motors.

