Lab 3: Radio-controlled Clock with DCF77

Dokumentation

Tim Jauch

Ergün Bickici

Contents

List of Figures

List of Tables

1 Requirements

- The first line of the Dragon12's LCD shall display the time in the format: `hh:mm:ss`. The second line shall show the date in the format: `dd.mm.yyyy`. Manual time setting, as implemented in Lab 2, is no longer required.
- Since DCF77 signal reception may occasionally fail (e.g., due to poor radio conditions in reinforced concrete buildings or during transmitter maintenance), the internal timer-based clock from Lab 2 remains active. The DCF77 signal is only used to adjust the time once per minute and to display the date. If the DCF77 data becomes unavailable, the system continues running solely on the internal timer.
- The internal timer clock shall toggle the LED connected to Port B.0 once per second.
- The LED on Port B.1 shall reflect the raw DCF77 signal: it turns on when the signal is low, and off when it is high. As a result, this LED will toggle once per second when the DCF77 signal is being received.
- The LED on Port B.3 shall be turned on when a complete and valid DCF77 time and date message has been successfully decoded. It turns off immediately if no valid data is received or the data is corrupted.
- The DCF77 signal is connected to Port H.0 on the real Dragon12 board. Due to potential signal spikes and jittering edges, the Signal is polled every 10 ms instead of relying on Port H interrupts.

2 User Interface Description

2.1 LCD Display

- The first line of the Dragon12 LCD displays the current time in the format `hh:mm:ss`.
- The second line shows the date in the format `dd.mm.yyyy`.
- The day of the week (e.g., `Sun`, `Mon`, ...) is displayed in front of the date.
- The active time zone is indicated by either `DE` or `US` on the LCD.

2.2 LED Indicators

- The LED connected to Port `B.0` toggles once per second, driven by the internal timer clock.
- The LED on Port `B.1` reflects the raw DCF77 signal level: it turns on when the signal is low and off when the signal is high. This results in a 1 Hz blinking pattern during proper DCF77 reception.
- The LED on Port `B.2` lights up when an error is detected and remains on until valid data is received.
- The LED on Port `B.3` turns on when a complete and valid DCF77 time and date frame is successfully decoded. It turns off immediately if the received data is invalid or missing.

2.3 Control Button

- The button connected to `PTH3` can be used to switch between US (East Coast) and DE time.

3 Debugging

No debugging was required, as all provided code segments functioned correctly. Only the necessary extensions to the `dcf77.c` and `clock.c` files had to be implemented.

4 Data Dictionary

4.1 List of Global Variables

Module(s)	Variable	Type	Purpose
clock.c, main.c	clockEvent	enum	Defines clock event types (e.g., SET, TICK, ALARM)
clock.c, dcf77.c	dcf77TimeZone	enum	Indicates selected time zone: US or DE
clock.c, dcf77.c, main.c	dcf77Event	enum DCF77EVENT	Holds the last detected DCF77 signal event (e.g., VALIDONE, VALIDZERO)
dcf77.c, main.c	us_on	unsigned char	Indicates whether US time zone mode is active (non-zero) or not (zero)
dcf77.c, main.c	DataOk	int	Signals whether the received DCF77 data is valid (non-zero) or invalid (zero)

Table 1: Global variables used across modules

4.2 Hardware Resources

Module	HCS12 Hardware	Purpose
dcf77.c	Port H	Receives the DCF77 antenna signal via input pin
led.asm	Port B, Port J	Controls status LEDs (DCF77 signal, timer pulses, error states)
lcd.asm	Dragon12 LCD	Displays time, date, weekday, and time zone

Table 2: Hardware resources accessed by software modules

5 Module overview

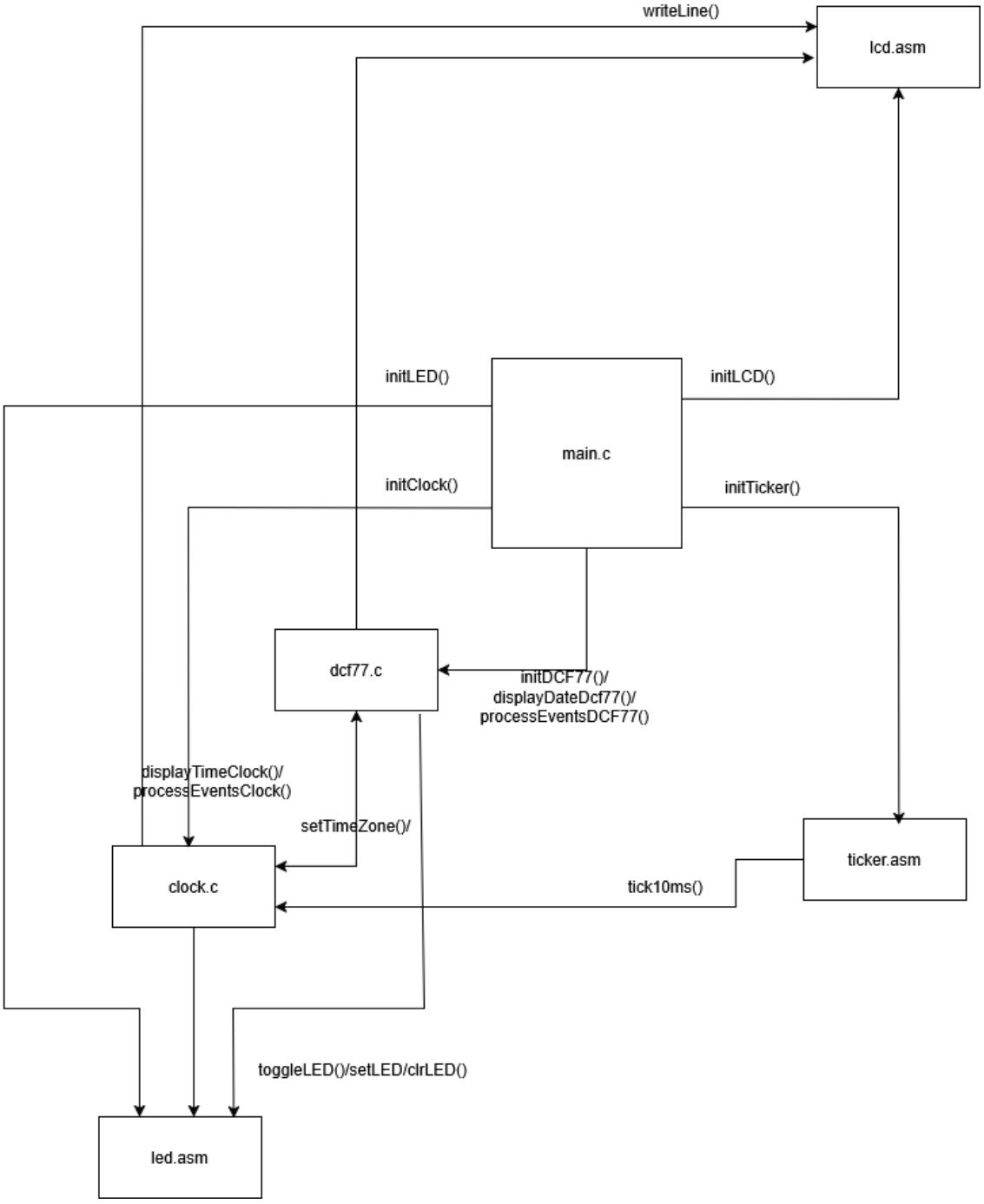

Figure 1: Module Overview

6 Interface Descriptions of All Subroutines

6.1 main.c

- **main()** Initializes all modules and calls the corresponding subroutines.

6.2 clock.c

- **initClock()** Initializes the clock module and displays the initial time on the LCD.
- **tick10ms()** Called every 10 ms by the system timer. Updates internal time base, detects full seconds, handles LED toggling, and samples the DCF77 signal.
- **processEventsClock(clockEvent)** Called once per second (on SECONDTICK). Advances the internal clock time (hours, minutes, seconds).
- **setClock(hours, minutes, seconds)** Sets the internal clock time. Resets the tick counter if seconds are set to 0.
- **displayTimeClock()** Displays the current time on the LCD. Prefixes with "US" or "DE" depending on the selected time zone.
- **time()** Returns the CPU uptime in milliseconds since system start.

6.3 dcf77.c

- **initializePort()** Configures Port H.0 as input to receive the DCF77 signal. Interrupts are disabled.
- **readPort()** Reads the signal level from Port H.0. Returns 0 if LOW, 1 if HIGH.
- **initDCF77()** Initializes the DCF77 module. Sets the initial clock and displays the decoded date on the LCD.
- **displayDateDcf77()** Formats and displays the decoded date on the second line of the LCD.
- **sampleSignalDCF77(currentTime)** Called every 10 ms to sample the DCF77 signal. Detects falling and rising edges and determines whether the signal represents a valid second, bit, or minute.
- **processEventsDCF77(event)** Processes DCF77 events. Stores received bits, handles complete minute frames, and triggers clock update and date display if valid data is received.
- **eventMinute()** Parses a full DCF77 minute frame. Extracts and validates date and time values, including parity checks. Updates internal variables if successful.

- **USDE()** Applies a `+6h` or `-6h` offset to switch between DE and US time zones. Adjusts weekday and date accordingly.
- **USDE_Pressed()** Checks if the button on Port H.3 is pressed. Toggles the selected time zone and updates display and internal time accordingly.
- **isLeap(year)** Determines whether the given year is a leap year. Returns 1 if true, 0 otherwise.
- **daysInMonth(month, year)** Returns the number of days in the specified month. Accounts for leap years in February.
- **adjustDate(day, month, year, deltaDays)** Adds or subtracts days to a given date, correctly wrapping over month and year boundaries. Used when switching time zones across midnight.

6.4 lcd.asm

- **initLCD()** Initializes the LCD display. Configures ports, sends initialization command sequences, and clears the display. Must be called once before using the display.
- **writeLine(X, B)** Writes a zero-terminated string to the specified LCD row. `X` is a pointer to the string, and `B` is the row number (0 or 1). If the string is shorter than 16 characters, the remaining space is filled with blanks. Longer strings are truncated.
- **delay_10ms()** Delays execution for approximately 10 milliseconds. Used for timing synchronization during LCD operations.

6.5 led.asm

- **initLED()** Initializes the LED output pins. Configures Port B as output and turns off all LEDs. Also activates LED control by enabling Port J.1 and (optionally) disables the seven-segment display.
- **toggleLED(bitmask)** Toggles the state of LEDs specified by the bitmask. A bit set to 1 in the mask inverts the current LED state.
- **setLED(bitmask)** Turns on the LEDs specified by the bitmask. Only bits set to 1 in the bitmask are affected.
- **clrLED(bitmask)** Turns off the LEDs specified by the bitmask. Only bits set to 1 in the bitmask are affected.

7 Flow Charts

7.1 Main

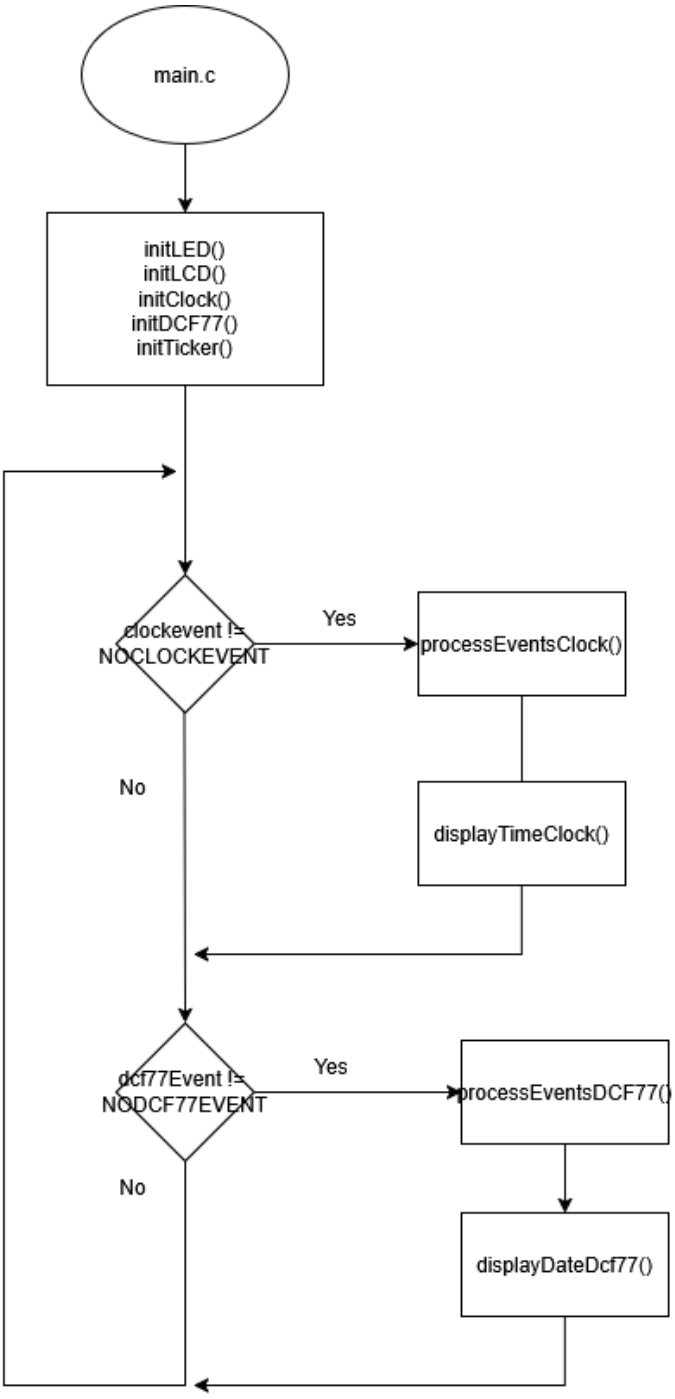

Figure 2: Main

7.2 Clock

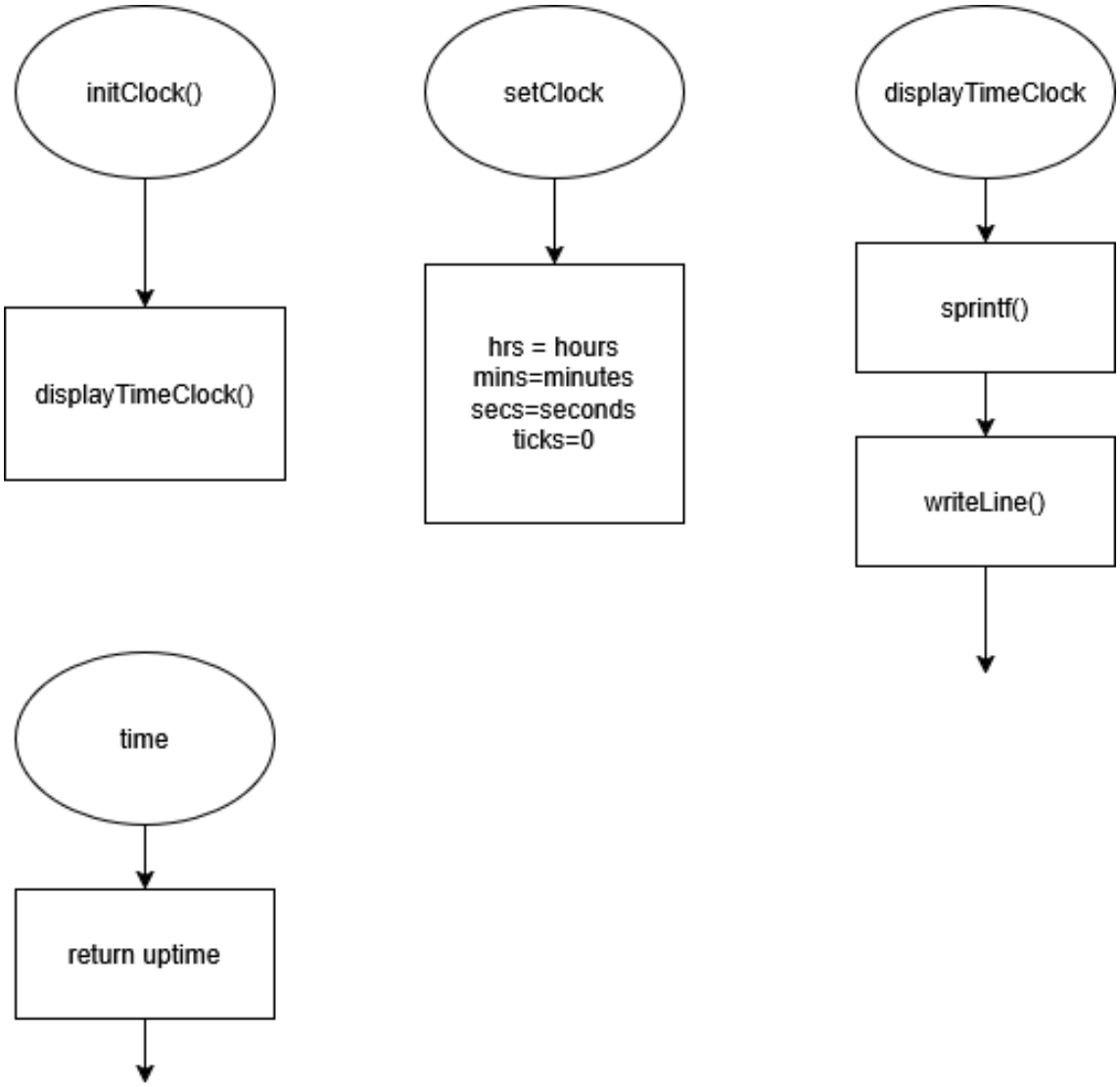

Figure 3: Clock 1

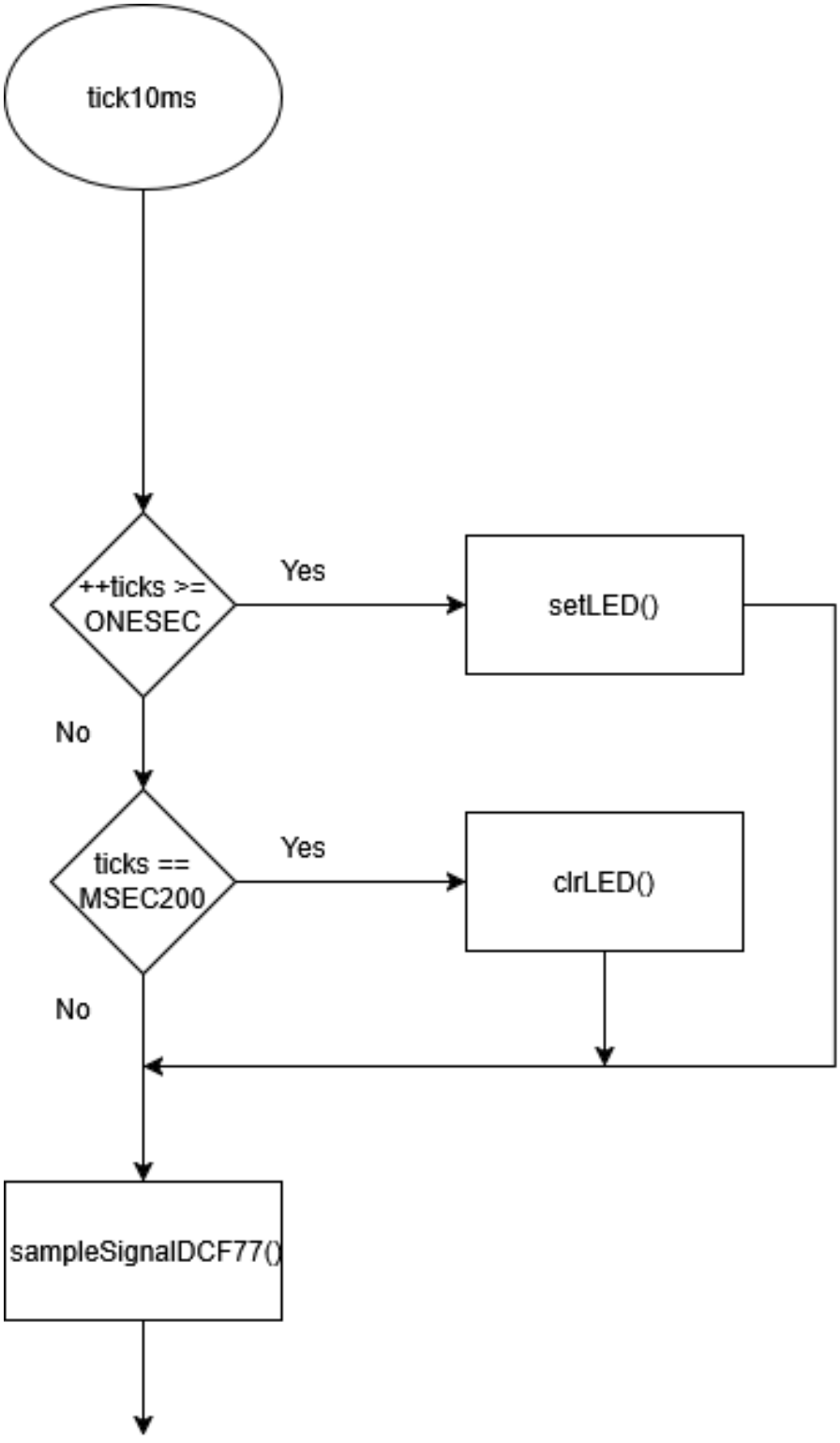

Figure 4: Clock 2

7.3 DCF77

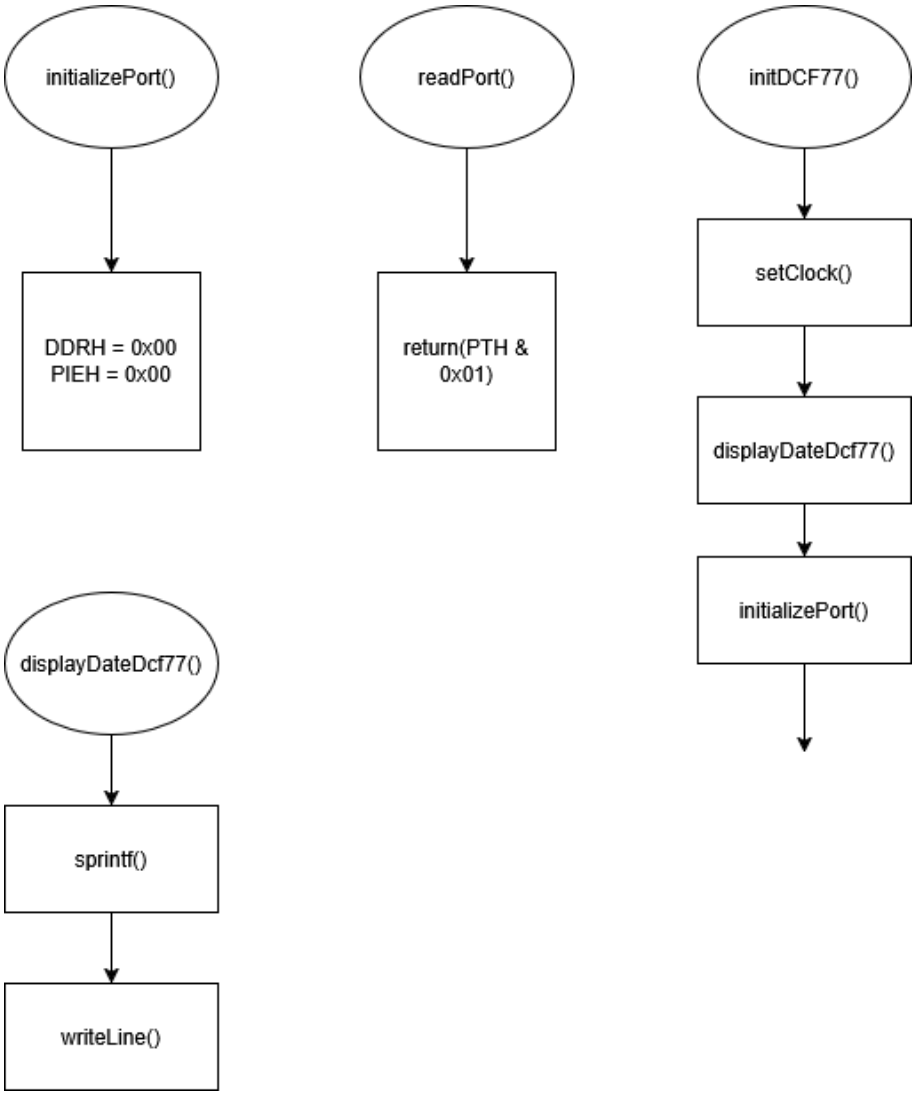

Figure 5: DCF77 - Part 1

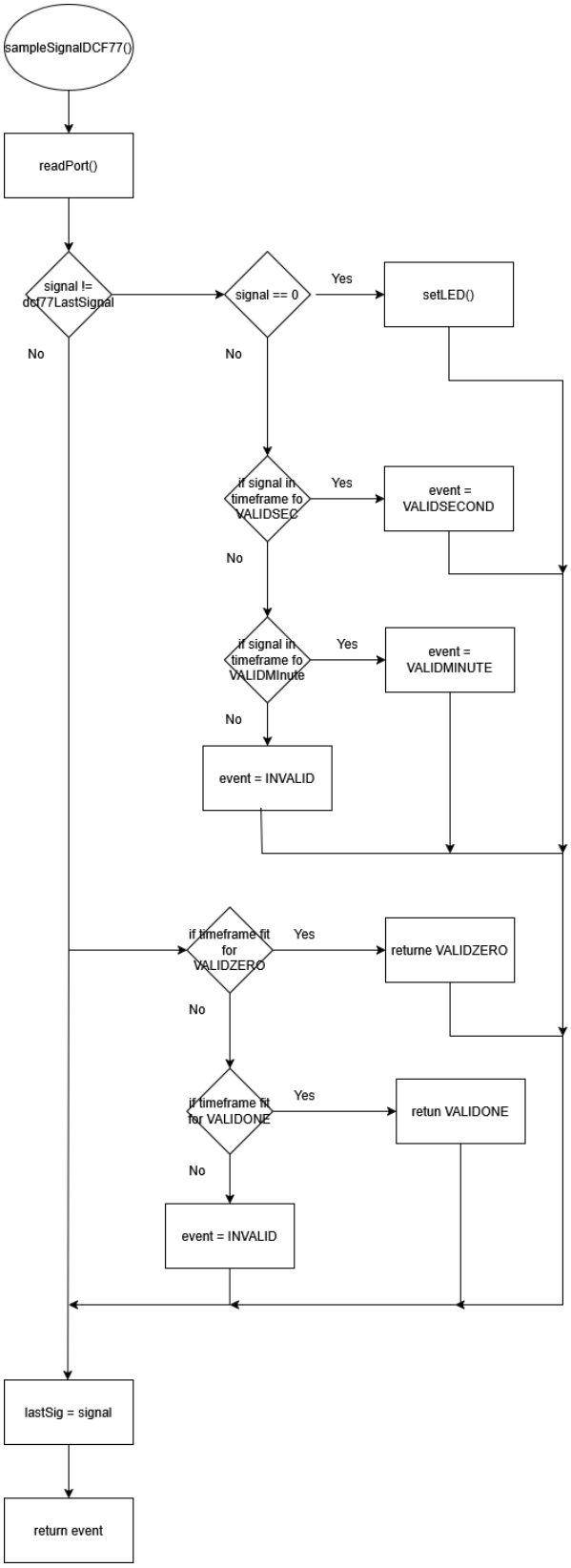

Figure 6: DCF77 - Part 2

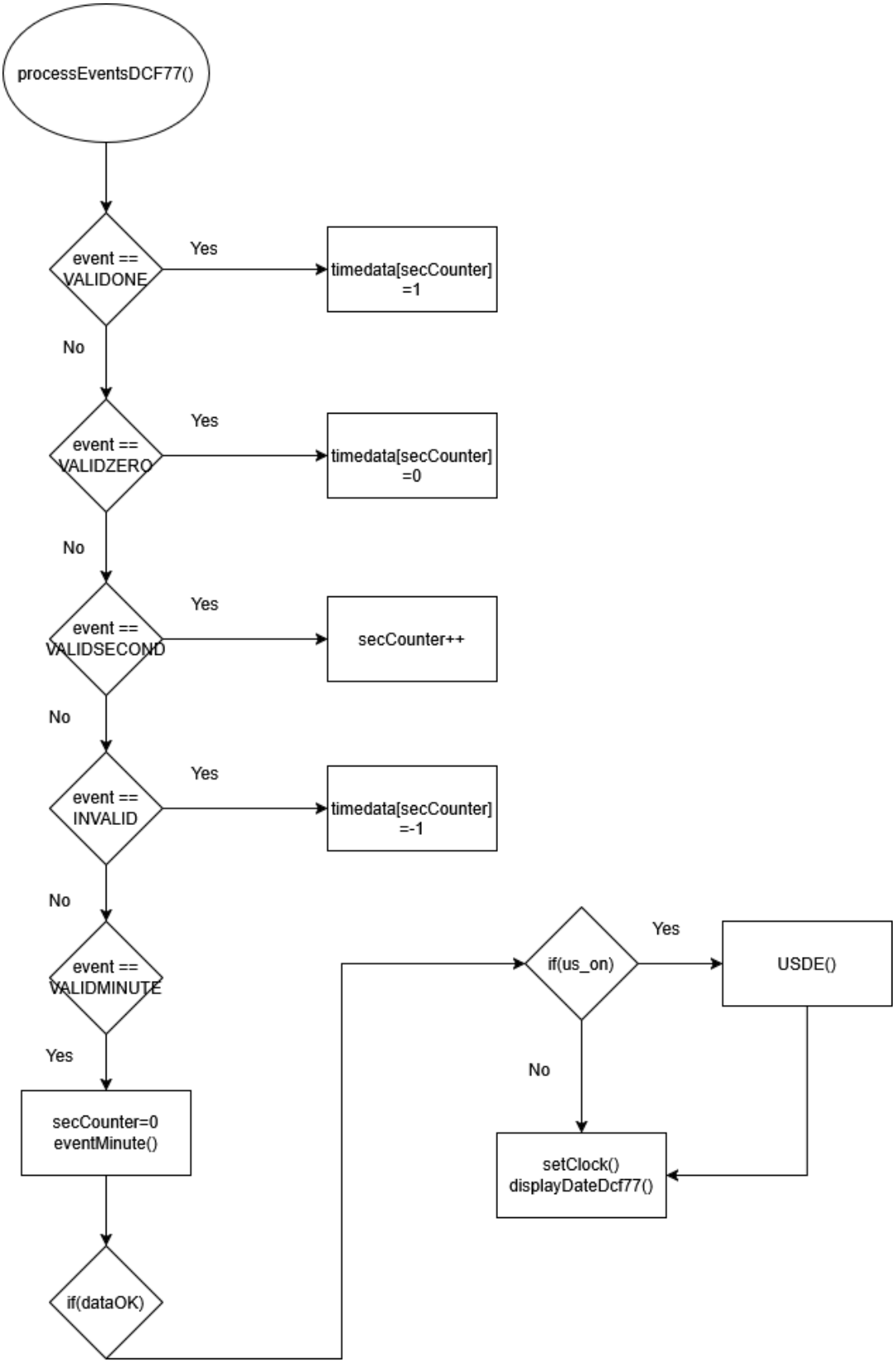

Figure 7: DCF77 - Part 3

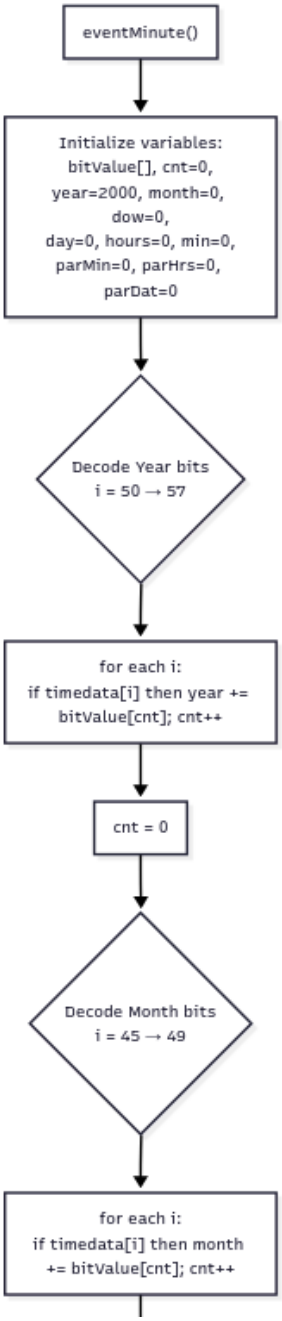

Figure 8: Event Minute – Part 1

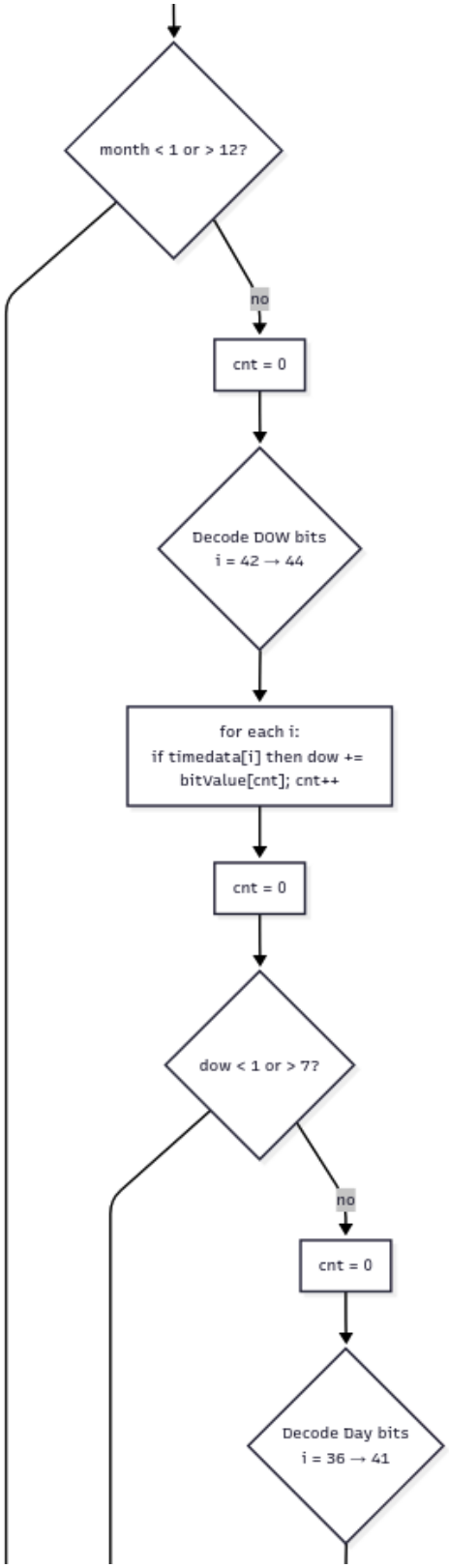

Figure 9: Event Minute – Part 2

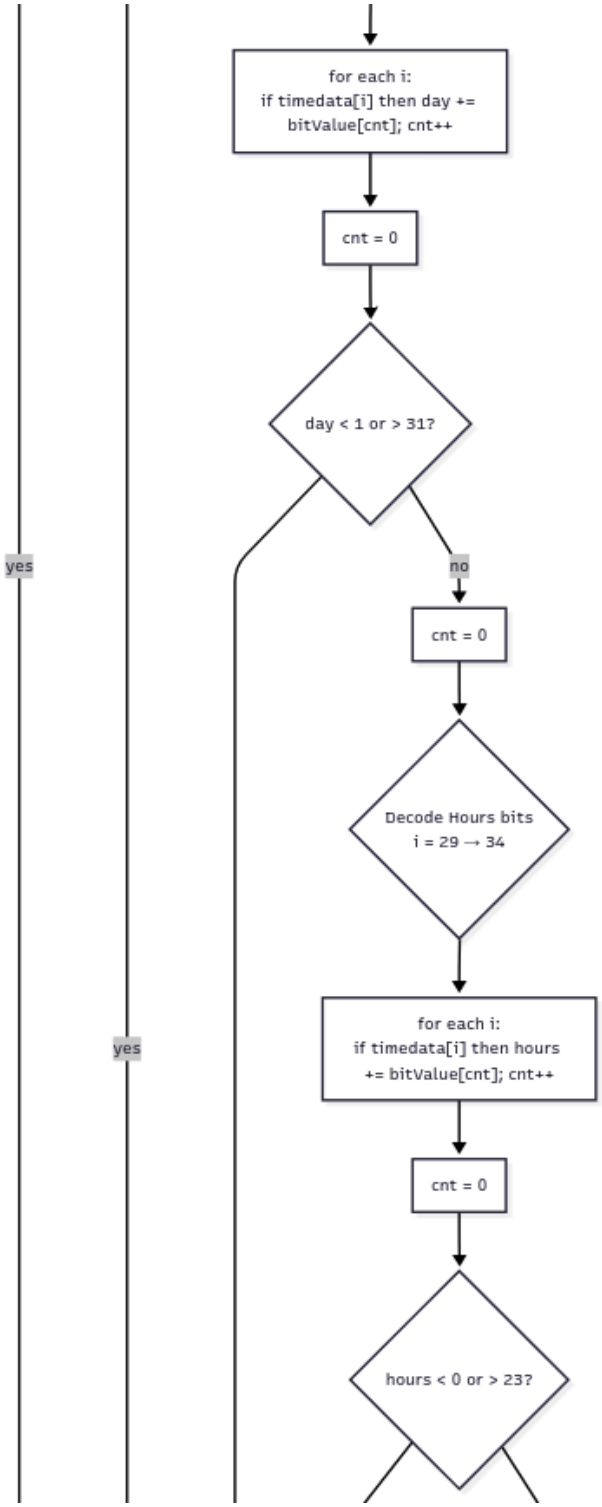

Figure 10: Event Minute – Part 3

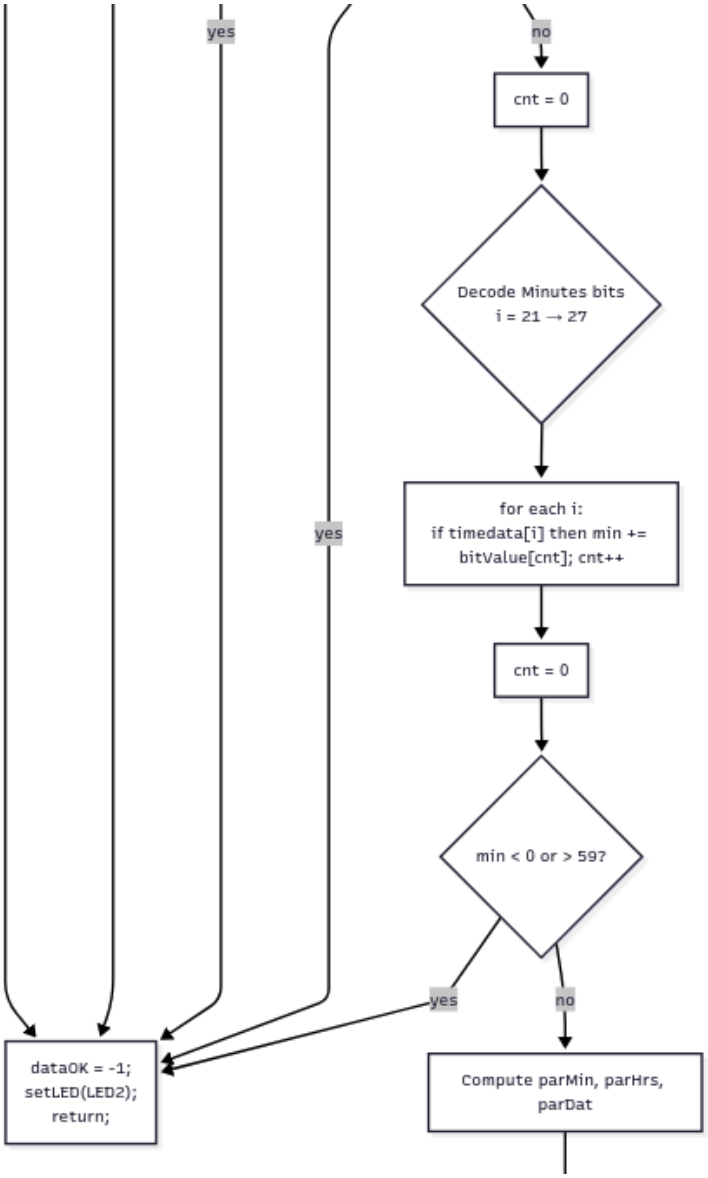

Figure 11: Event Minute – Part 4

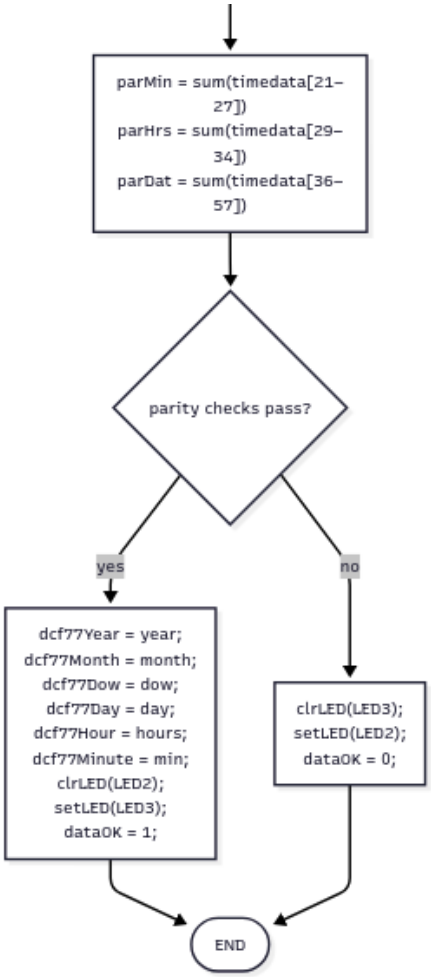

Figure 12: Event Minute – Part 5

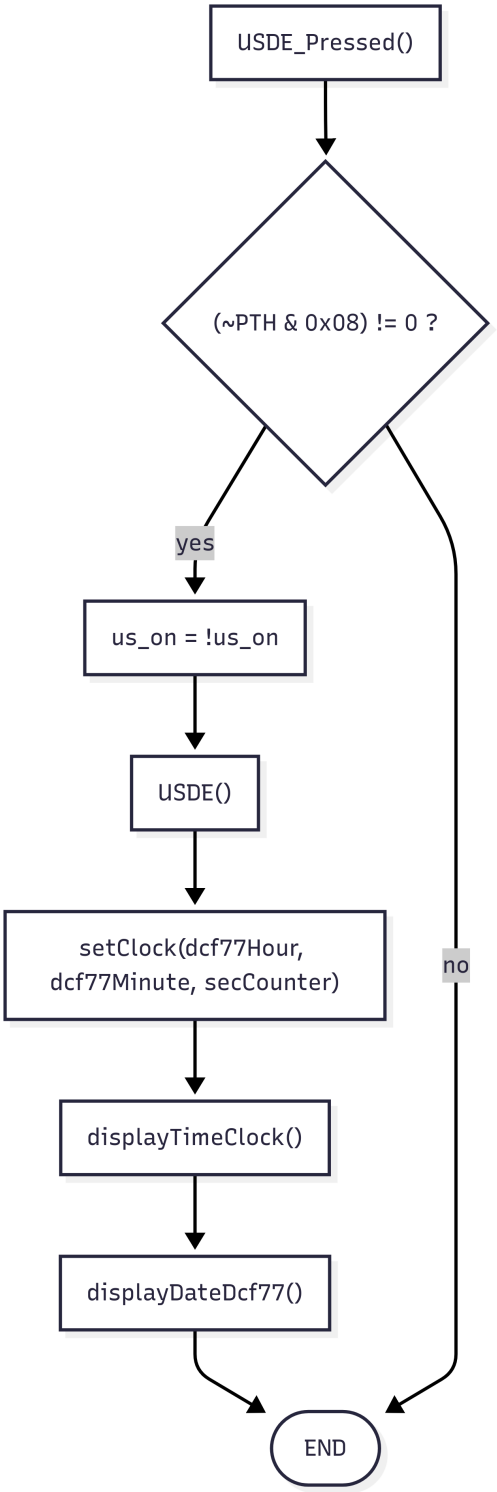

Figure 13: USDE Pressed

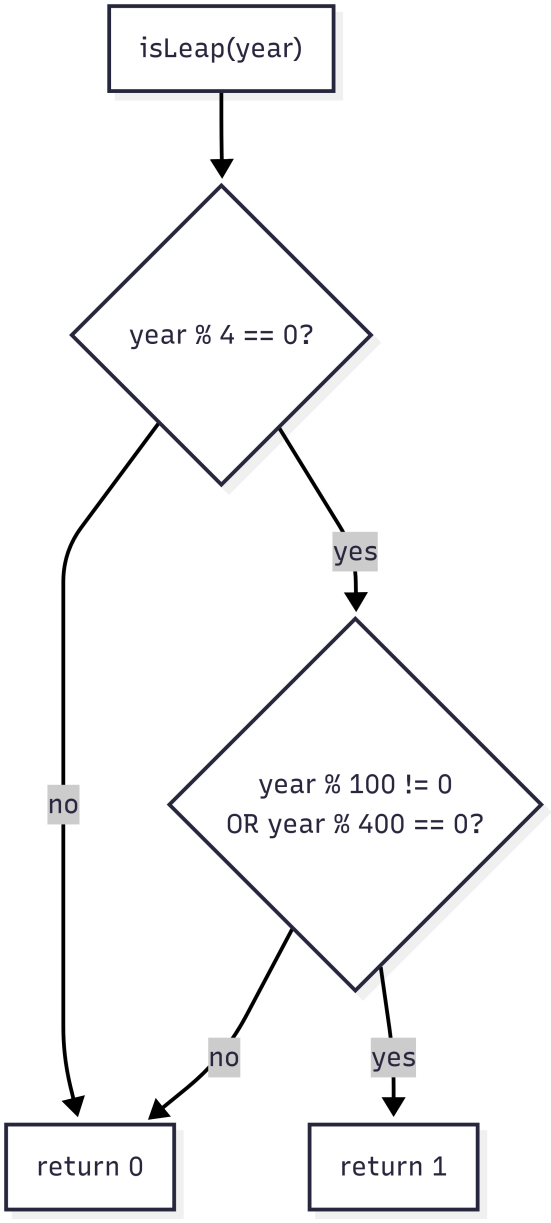

Figure 14: isLeap

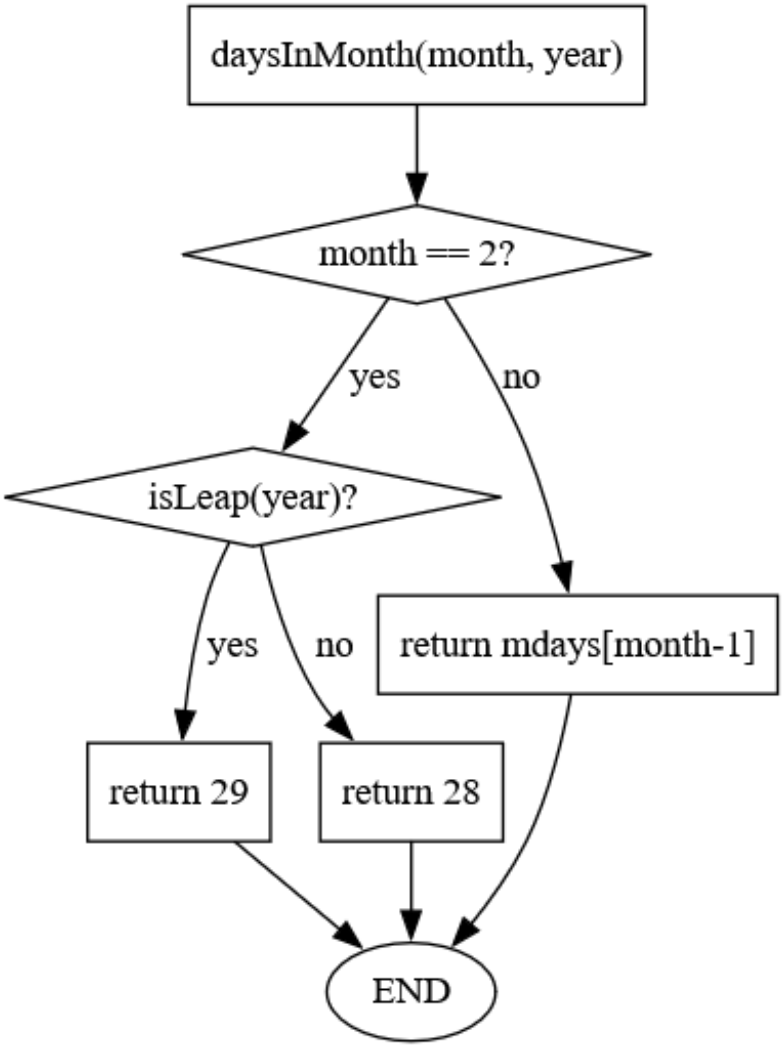

Figure 15: daysInMonth

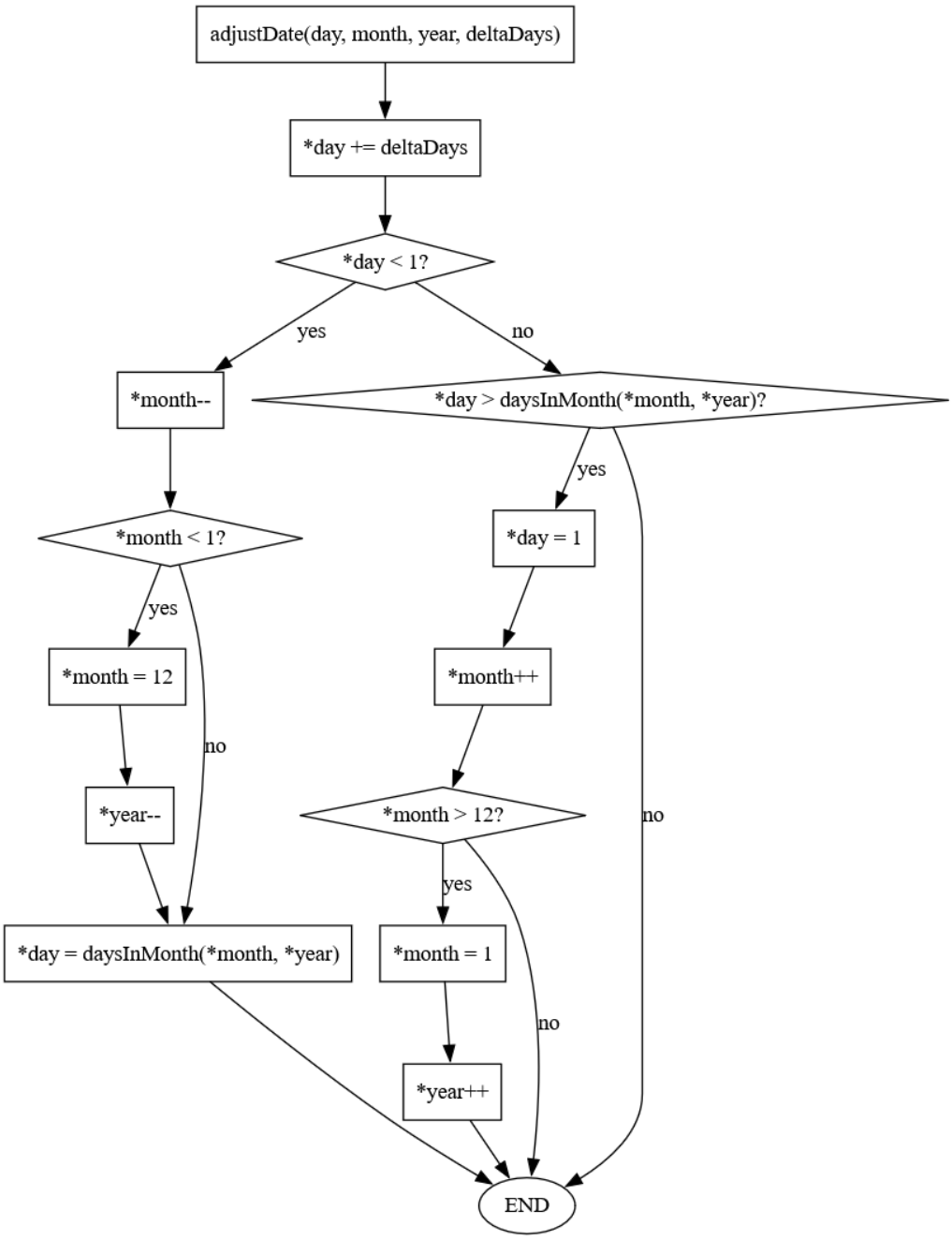

Figure 16: adjustDate

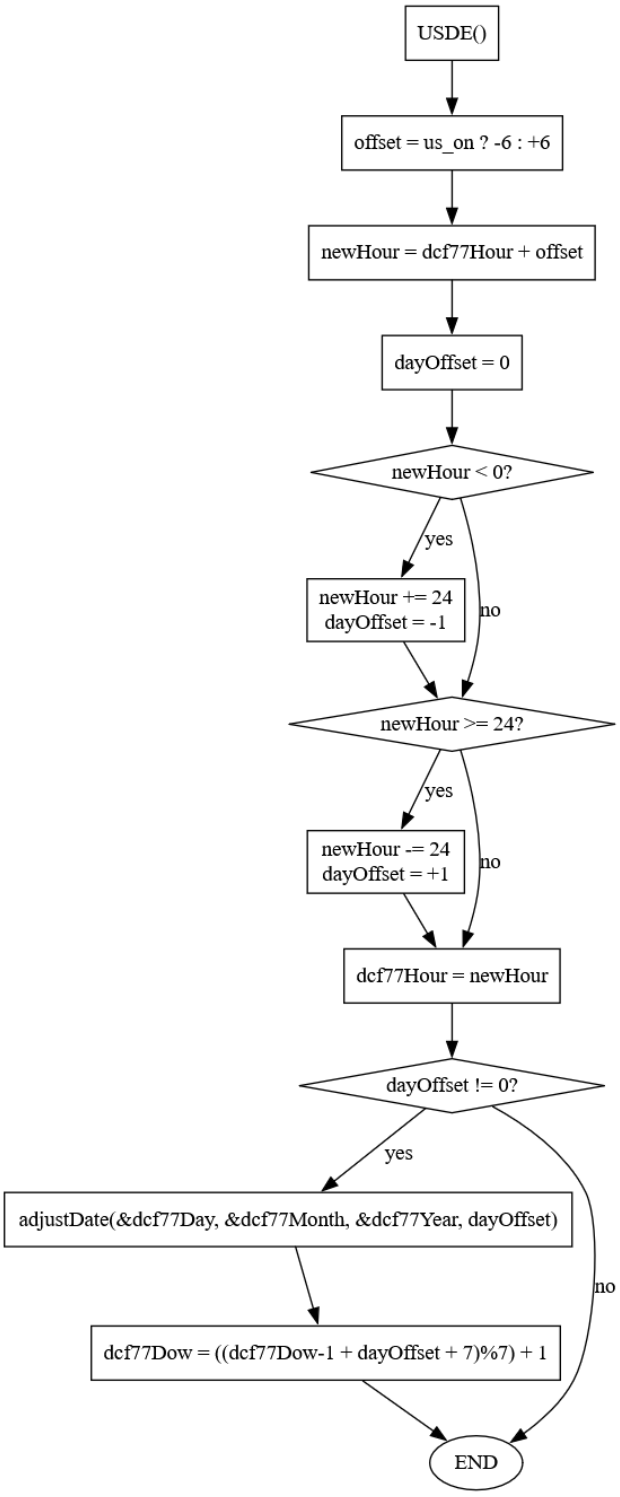

Figure 17: USDE