CMPE 663 Project 1

Timing

Author: Gloria Mbaka
Email: gm6629@rit.edu

Submitted: 09/18/24

Instructor: Prof. Wolfe

TA: Bertola Thomas

Analysis/Design

Project Overview

This project involved the development of a bare-metal embedded system on the STM32 Nucleo board to measure and process rising-edge pulse inter-arrival times. The system aimed to categorize pulse inter-arrival times into user-defined time buckets, with configurable lower and upper limits. By default, the limits were set to 950 µs and 1050 µs, but the user could change these values through a terminal interface (PuTTY).

The core functionality of the system was implemented using STM32's TIM2 timer peripheral in input capture mode**,** with results displayed through the UART terminal. The system captures 1000 pulse inter-arrival times (between 1001 rising edges), processes them into 1-microsecond-wide time buckets, and displays the non-zero results in ascending order.

Software Design Approach

The project was divided into two key phases:

1. Initialization:
    - Timer Initialization: TIM2 was configured in input capture mode to detect rising edges and capture the time of each event.
    - UART Initialization: UART was set up to allow communication with a terminal (PuTTY) for displaying results and accepting user inputs.
    - GPIO Setup: GPIO was used to capture external pulse signals.
2. Main Execution Loop:
    - Power-On Self-Test (POST): Upon startup, the system performs a POST to ensure pulse detection is functional before capturing any data.
    - User Configuration: The user can accept default time limits or input new values for the lower and upper limits.
    - Pulse Capture: The system captures 1000 inter-arrival times between pulses and groups them into buckets representing 1-microsecond intervals.
    - Results Display: After processing, the results are displayed in a two-column format showing the time (in microseconds) and the count of pulses within that time range.

Software Flow:

The main execution flow repeats after each set of 1000 measurements, allowing the user to change the time limits or run the same test again.

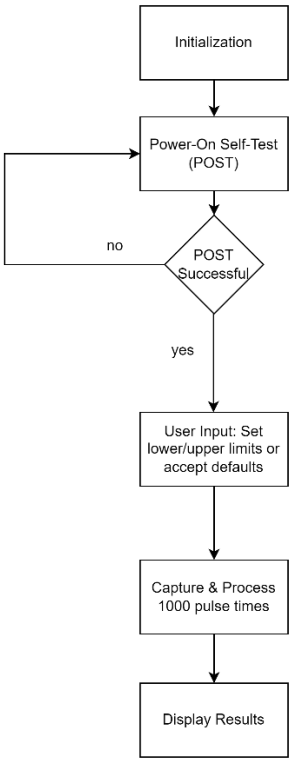

Figure 1: Software Design Flow Chart

Hardware Design

- STM32 Nucleo Board**:**
  - Timer Peripheral (TIM2)**:** Configured for input capture mode to track rising edges of an external pulse signal.
  - UART (USART2)**:** Handles serial communication with the PuTTY terminal for user input and output.
  - GPIO Pins**:** Used to capture pulse signals.

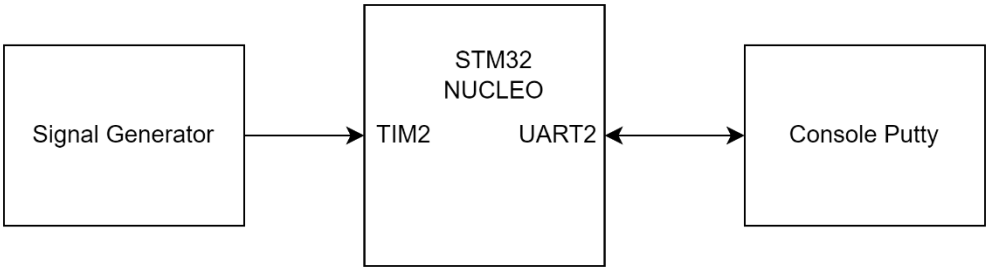

Figure 2: Hardware Block Diagram

Test Plan

Power-On Self-Test (POST):
Upon startup, the system performs a POST to verify that pulses are being detected. This is done by starting the timer in input capture mode and waiting for a rising-edge event within 100 milliseconds. If no pulse is detected, the user is prompted to retry. If a signal is detected, the system proceeds to the user interface, allowing further configurations.

Additional Tests:

- User Input Validation: After POST, the user is prompted to input the lower time limit (between 50 µs and 9950 µs). The upper limit is automatically set 100 µs higher than the lower limit. If the user input is invalid, the system prompts again for valid input.
- Pulse Capture Test**:** The system was tested using an oscilloscope signal generator (1 kHz) and an external signal generator to ensure accurate pulse capture and processing.
- Results Display**:** After capturing 1000 pulse inter-arrival times, the program outputs the results in a table format, showing the non-zero counts for each time bucket. The display format includes the time in microseconds (starting from the lower limit) and the count of pulses that fall within that time range.

Expected Results:

- POST should succeed or provide the option to retry.
- Lower limit should default to 950 µs, and upper limit to 1050 µs.
- Captured pulses should fill appropriate time buckets (increments of 1 µs), with non-zero results displayed correctly.

Project Results

Test 1: Default Settings (Lower Limit: 950 µs, Upper Limit: 1050 µs)

Expected Results:

- The system should capture 1000 pulse inter-arrival times.
- The majority of pulse times should fall within the 1000 µs bucket when using a 1 kHz signal.
- Only non-zero bucket counts should be printed.

The result of the test 1 is shown in figure 3. For this test, the system used the default time limits where the lower limit was set to 950 µs and the upper limit was set to 1050 µs. The purpose of this test was to verify that the system correctly captures inter-arrival times of pulses and categorizes them into 1-microsecond-wide buckets.

- Key Observations:
  - The inter-arrival times clustered predominantly around the 1000 µs range, as expected for a pulse source with a frequency close to 1 kHz.
  - The buckets between 999 µs and 1001 µs contain the majority of pulse counts, with 1000 µs showing the highest count of 481 pulses.
  - There are smaller counts for slightly off-target inter-arrival times at 997 µs and 1002 µs, which likely reflect minor variations in the timing of the pulses.

Thus, the system accurately captured pulses within the defined range and correctly placed them into the appropriate buckets, as expected for a 1 kHz input signal.

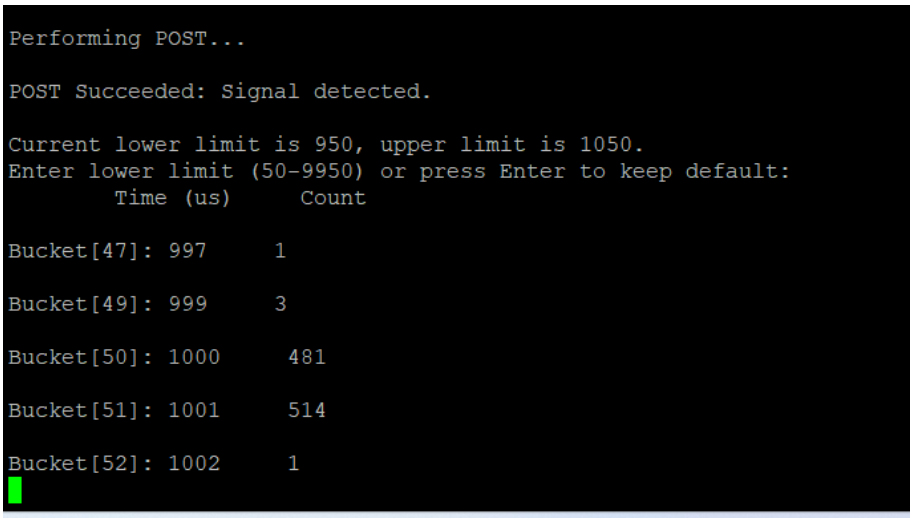

Figure 3: Test Result from Default setting

Test 2: Modified Limits (Lower Limit: 2000 µs, Upper Limit: 2100 µs)

Expected Results:

- When the lower limit is changed to 2000 µs, the system should capture pulse inter-arrival times in the range of 2000 µs to 2100 µs, and the results should reflect these limits.
- Non-zero counts should be displayed for each time bucket where pulse inter-arrival times fall.

The result of test 2 is shown in figure 4. In this test, the lower time limit was modified to 2000 µs, and the upper limit was adjusted automatically to 2100 µs. This test was performed to check whether the system adapts correctly to new time limits set by the user.

- Key Observations:
  - The pulse counts are concentrated around the 2000 µs range, with the bucket for 2000 µs showing the highest count of 630 pulses.
  - Fewer pulses were recorded in the subsequent time intervals, with 344 pulses in the 2001 µs bucket and 26 pulses in the 2002 µs bucket. This indicates that most inter-arrival times fell closer to the lower limit.

Thus, the system successfully adapted to the modified time limits, capturing and categorizing pulse inter-arrival times within the 2000–2100 µs range. This demonstrates the flexibility of the system in handling user-defined configurations.

```
POST Succeeded: Signal detected.

Current lower limit is 950, upper limit is 1050.
Enter lower limit (50-9950) or press Enter to keep default: 2000
New limits set: lower limit = 2000, upper limit = 2100

        Time (us)     Count

Bucket[0]: 2000     630

Bucket[1]: 2001     344

Bucket[2]: 2002     26
```

Figure 4: Test Result from Modified Settings

Comparison of 1 kHz Results Using an Oscilloscope and Signal Generator:

When using the oscilloscope signal at 1 kHz, the captured results closely matched the expected 1000 µs interval for inter-arrival times, confirming that the timer peripheral was correctly configured and functioning as expected. The results from the external signal generator were consistent with the oscilloscope test, confirming accurate pulse detection.

Lessons Learnt

Key Takeaways:

- Peripheral Handling**:** The project provided hands-on experience with configuring and using the STM32 timer peripheral for input capture, as well as UART for serial communication. Managing these peripherals in a bare-metal environment was a key learning experience.
- Real-Time Signal Processing**:** Working with real-time signal capture and processing under interrupt-driven conditions emphasized the importance of efficient interrupt handling and flag management in embedded systems.
- User Interface Design**:** Implementing a user interface through a terminal (PuTTY) and handling user input validation added complexity to the project. The experience of building a robust and interactive interface for embedded systems was valuable.

Challenges Encountered:

One of the main challenges was ensuring that the program accurately captured a total of 1000 pulse inter-arrival times. Initially, the system was recording more than 1000 pulses, which required debugging the flag management in the interrupt handler and ensuring that the program properly resets and increments the counters. This issue was resolved by refining the interrupt service routine and adjusting the conditions for when to stop counting pulses.