STM32G491RE: MoT Peripheral Integration User Manual

Author: Andreas Tzitzikas
Version: 1.0
Release Date: 05/19/2025

Introduction

This user manual provides operating instructions for selected peripherals of the STM32G491RE microprocessor, controlled via the Microcontroller on Tether (MoT) system developed for the ENEE 440 course. The software leverages the MoT communication framework, originally provided by Professor Hawking. Within this framework, Andreas Tzitzikas implemented the specific control modules for the peripherals detailed in this guide. This document will explain the necessary commands and procedures to effectively interact with and utilize these implemented functionalities.

Table of Contents

Introduction..... 1
Table of Contents..... 2
MoT System Overview..... 4
Introduction..... 4
Using testterm.exe..... 4
Prerequisites..... 4
Test Script Format..... 4
MoT Command String Format..... 4
Running testterm.exe..... 5
Interactions..... 5
Device 3: General Purpose Input/Output (GPIO)..... 6
Introduction..... 6
Input Commands..... 6
Initialize..... 6
Immediate Read..... 6
Scheduled Read..... 7
Monitored Read..... 8
Initialize External Trigger (EXTI1)..... 8
Stop Monitor..... 9
Return Triggered Results..... 10
Output Commands..... 10
Immediate Low..... 10
Immediate High..... 11
Scheduled..... 11
Repetitive..... 12
Device 4: Digital-to-Analog Converter (DAC)..... 14
Introduction..... 14
Input Commands..... 14
Initialize DAC..... 14
Enable External Trigger..... 14
Output Commands..... 15
Set Constant Voltage..... 15
Start Waveform Output..... 16
Stop Waveform Output..... 17
Device 5: Analog-to-Digital Converter (ADC)..... 18
Introduction..... 18
Input Commands..... 18
Initialize ADC..... 18
Report ADC Value..... 18
Report if ADC Value is Above/Below Thresholds..... 19
Stop Monitor Task..... 20

Initialize External Trigger (EXTI1)..... 20
Device 6: TIMER (TIM2).....22
Introduction.....22
Output Commands.....22
PWM On (Pulse Width Modulation)..... 22
PFM On (Pulse Frequency Modulation)..... 23
Simple Pulse On..... 23
Pulse Width On..... 24
Pulse Frequency On..... 25
Initialize Trigger..... 25
Triggered Control.....26
Timer Function Off.....27
Device 7: SPI Loopback Testing.....28
Introduction.....28
Input Commands.....28
Initialize SPI..... 28
SPI Write.....28
Clear SPI Buffers..... 29
Output Commands.....30
SPI Read..... 30
Device 8: W25QXX Flash Memory Storage..... 30
Introduction..... 30
Control Commands..... 31
Reset/Initialization.....31
Erase.....31
Copy RAM Buffer to Flash..... 32
Verify Flash against RAM Buffer..... 32
Input Commands.....33
Input Data to RAM Buffer.....33
Output Commands.....34
Read JDEC.....34
Output Data from Flash.....34

MoT System Overview

Introduction

The Microcontroller on Tether (MoT) system is a framework designed for interacting with and controlling peripherals on the STM32G491RE microprocessor. This system, originally developed by Professor Hawking and with specific peripheral control modules implemented by Andreas Tzitzikas, allows users to send commands from a host computer to the microcontroller to perform various operations, such as initializing hardware, setting outputs, or reading inputs.

Users interact with the MoT system primarily through the testterm.exe program, a custom serial terminal. This program facilitates sending command scripts from the host PC to the STM32G491RE board

Using testterm.exe

Prerequisites

There are three things needed to operate the testterm.exe. First find the COM port to which the Nucleo board is connected to and create a test script file whose format can be found below and within the test scripts provided. At last ensure that the board is connected and the provided code is running in either debug or run mode.

Test Script Format

File: .txt

Comment lines: Lines starting with a # in the first column are considered comments. These are displayed by testterm.exe for information purposes but are not sent to the MoT device.

Command lines: Lines starting with : in the first column are treated as commands to be sent to the MoT device.

MoT Command String Format

Commands follow a specific structure: :[DeviceID][CmdID][Data...][CS]

- : : Start character for a command.
- DeviceID : A two-digit hexadecimal number identifying the target MoT device (e.g., "04" for Device 4)
- CmdID : A two-digit hexadecimal number specifying the command for the target device (e.g., "00" for Initialize Device).
- Data… : Optional hexadecimal data bytes required by the specific command.
- CS : A two-digit hexadecimal checksum. The checksum is calculated as (0 - (sum of DeviceID + CmdID + DataBytes)) & 0xFF

Running testterm.exe

Launch testterm.exe from a command line, providing the COM port and the script filename as arguments.)e.g., testterm.exe COM5 test_scipt.txt

Interactions

During operation, testterm.exe will present informational messages from the script—lines beginning with '#'—one at a time. To move through these messages or to execute a command, the user is required to press 'Enter'.

When a command line is reached in the script, pressing 'Enter' prompts testterm.exe to transmit the command string to the MoT device over the serial port. The console will then confirm this action by displaying the number of bytes sent along with the transmitted message, for example, txmsg= :0100FF.

It is crucial to wait for a "bytes received" confirmation from testterm.exe after a command is sent before advancing to the next step in the script to ensure smooth operation. This confirmation will appear in the console, indicating the number of bytes received and the actual response message from the MoT device, such as rxmsg= userLED is initialized.

Device 3: General Purpose Input/Output (GPIO)

Introduction

Device 3 offers comprehensive control over General Purpose Input/Output (GPIO) functionalities, primarily centered around pin PA8. It allows for configuring PA8 as either a digital input to read its state, or as a digital output to set its state. Input operations include immediate reads, scheduled reads after a specified delay, and continuous monitoring for state changes. Furthermore, Device 3 can utilize pin PA1 as an external trigger source to capture the state of PA8 upon an event. For output, PA8 can be set HIGH or LOW immediately, after a delay, or used to generate repetitive square wave signals.

Commands are sent to Device 3 as part of a MoT message. The first byte of the payload for Device 3 should contain the command code.

Input Commands

Initialize

Command: 0x00

Description: This command initializes Device 03 and configures its primary GPIO pin (PA8) for input operations. The fundamental setup performed by the initialization routine includes enabling the clock for GPIO Port A and configuring PA8 without pull-up or pull-down resistors, making it a floating input. It must be called before other Device 03 commands that rely on PA8 being an initialized input (such as reading its state or monitoring it for changes) can be successfully executed.

Parameters: None.

Payload Structure:

- 00: Command Code for Initialize GPIO.

Confirmation Message: Upon successful initialization, the device will post the message: “device3 has been initialized” to the console

MoT Command String Example:
:0300FD

- : : Start character
- 03 : Device 03 ID
- 00 : Command Code for Initialize Device 03
- FD : Checksum (calculated for the string "0300")

Console Response:
device3 has been initialized

Immediate Read

Command: 0x05

Description: This command reads the immediate digital state of GPIO pin PA8 on Device 03. Upon execution, it first ensures PA8 is configured as a floating input (no pull-up or pull-down resistors). It then samples the current logic level of PA8 and reports to the console whether the pin is 'HIGH' or 'LOW'. For accurate readings, the external logic level on PA8 should be established and stable *before* this command is issued, as noted in the test script guidance for this function.

Parameters: None.

Payload Structure:

- 05 : Command Code for Read Immediate Input.

Confirmation Message: Upon successful initialization, the device will post the message: "Input is HIGH" or "Input is LOW" to the console

MoT Command String Example: :0305F8

- : : Start character
- 03 : Device 03 ID
- 05 : Command Code for Read Immediate Input
- F8 : Checksum (calculated for the string "0305")

Console Response (Example if PA8 is HIGH):
Input is HIGH

Scheduled Read

Command: 0x06

Description: This command reads the digital state of GPIO pin PA8 on Device 03 after a user-specified delay, which is provided as a parameter. Upon execution, it first ensures PA8 is configured as a floating input (no pull-up or pull-down resistors). The device then posts an initial "Waiting on PA8" message to the console. Following the specified delay period, it samples the current logic level of PA8 and reports to the console whether the pin is subsequently found to be 'HIGH' or 'LOW'. For accurate readings, the external logic level on PA8 should be established and stable at the actual time of sampling after the delay.

Parameters:

- Delay (16-bit): An unsigned integer representing the duration to wait before reading the state of pin PA8. This value is provided as two bytes in the command's data field, representing a 16-bit hexadecimal number.

Payload Structure:

- 06 : Command Code for Read Scheduled Input
- DH DL : DH (Delay High Byte) and DL (Delay Low Byte) are two hexadecimal bytes that together represent the 16-bit delay duration.

Confirmation Message: Upon successful initialization, the device will post the message: "Waiting on PA8" to the console. After the specified delay and the pin read, the console will display the input state, for example: "Input is HIGH" or "Input is LOW".

MoT Command String Example:
:030600FAFD

- : : Start character
- 03 : Device 03 ID
- 06 : Command Code for Read Scheduled Input
- 00FA : Delay value (High Byte 00, Low Byte FA = 250 ms)
- FD : Checksum (calculated for the string "030600FA")

Console Response (If PA8 is connected to HIGH):
Waiting on PA8
Input is HIGH

Monitored Read

Command: 0x07

Description: This command initiates continuous monitoring of the digital state of GPIO pin PA8 on Device 03. Upon execution, it first ensures PA8 is configured as a floating input (no pull-up or pull-down resistors) and records its initial state. The device then posts an initial "Monitor on PA8" message to the console. Subsequently, a background task periodically samples PA8. This monitoring continues until explicitly stopped by the stop monitor function and then the changes are outputed to the console.

Parameters: None.

Payload Structure:
- 07 : Command Code for Start Monitoring Input.

Confirmation Message:
Upon starting the monitor, the device will post "Monitor on PA8" to the console.

MoT Command String Example:
:0307F6
- : : Start character
- 03 : Device 03 ID
- 07 : Command Code for Start Monitoring Input
- F6 : Checksum (calculated for the string "0307")

Console Response:
Monitor on PA8

Initialize External Trigger (EXTI1)

Command: 0x08

Description: This command sets up an external interrupt (EXTI) system using pin PA1 as the trigger input. When an event occurs on PA1, the current state of another pin, PA8, is captured. First, PA8 is configured as a floating input, making it ready to be read. Next, the EXTI system is initialized for PA1. This setup allows PA1 to detect a rising or falling edge. Once configured, any external electrical event on PA1 will cause the system to capture the state of PA8 at that exact moment. This captured value can later be retrieved with return triggered results.

Parameters: None.

Payload Structure:
- 08 : Command Code for Arm External Trigger.

Confirmation Message: Upon successful arming of the trigger, the device will post "Trigger on PA1" to the console. This indicates that pin PA1 is now active as the trigger input.

MoT Command String Example:
:0308F5
- : : Start character
- 03 : Device 03 ID
- 08 : Command Code for Arm External Trigger
- F5 : Checksum (calculated for the string "0308")

Console Response:
Trigger on PA1

Stop Monitor

Command: 0x09

Description: This command stops the continuous monitoring of GPIO pin PA8 on Device 03, which was initiated by the "Start Monitoring Input" command (`0x07`). This action effectively halts any further sampling of PA8 and outputs reports of its state changes as "Change on PA8" followed by "Input is HIGH" or "Input is LOW". A confirmation message, "Monitor Has Stopped", is then posted to the console to confirm the cessation of monitoring activity.

Parameters: None.

Payload Structure:
- 09 : Command Code for Stop Monitoring PA8.

Confirmation Message:
Upon successful execution, if any changes occurred the device will post “Change on PA8” followed by the new state “Input is High” or “Input is Low” then the device will post "Monitor Has Stopped" to the console.

MoT Command String Example:
:0309F4
- : : Start character
- 03 : Device 03 ID
- 09 : Command Code for Stop Monitoring PA8
- F4 : Checksum (calculated for the string "0309")

Console Response (If PA8 is moved from LOW to HIGH after Monitor Command):
Change on PA8
Input is High
Monitor Has Stopped

Return Triggered Results

Command: 0x0A

Description: This command is intended to be used after an external trigger event has occurred. The trigger mechanism is set up by the "Initialize Trigger" command (0x08), where an event on pin PA1 is configured to capture the state of pin PA8. The purpose of this command is to output the state of PA8 after each trigger.

Parameters: None.

Payload Structure:

- 0A : Command Code for Read EXTI Trigger Results.

Confirmation Message: Upon execution, the device will post the states after each trigger as either “Input is HIGH” or “Input is LOW" and then post “Result from Trigger" to the console.

MoT Command String Example:
:030AF3

- : : Start character
- 03 : Device 03 ID
- 0A : Command Code for Read EXTI Trigger Results
- F3 : Checksum (calculated for the string "030A")

Console Response (If PA1 is triggered once and PA8 is on HIGH):
Input is HIGH
Result from Trigger

Output Commands

Immediate Low

Command: 0x01

Description: This command immediately sets the output state of GPIO pin PA8 on Device 03 to LOW. Upon execution, it first ensures PA8 is configured for output mode with no pull-up or pull-down resistors. It then actively drives the PA8 pin to a logic LOW state.

Parameters: None.

Payload Structure:

- 01 : Command Code for Set PA8 Output Low - Immediate.

Confirmation Message:
Upon execution, the device posts the message "Input is LOW" to the console as confirmation for this output operation

MoT Command String Example:
:0301FC

- : : Start character
- 03 : Device 03 ID

- 01 : Command Code for Set PA8 Output Low - Immediate
- FC : Checksum (calculated for the string "0301")

Console Response:
Input is LOW

Immediate High

Command: 0x02

Description: This command immediately sets the output state of GPIO pin PA8 on Device 03 to HIGH. Upon execution, it first ensures PA8 is configured for output mode with no pull-up or pull-down resistors. It then actively drives the PA8 pin to a logic HIGH state.

Parameters: None.

Payload Structure:
- 02 : Command Code for Set PA8 Output High - Immediate.

Confirmation Message: Upon execution, the device posts the message "Input is HIGH" to the console as confirmation for this output operation.

MoT Command String Example:
:0302FB
- : : Start character
- 03 : Device 03 ID
- 02 : Command Code for Set PA8 Output High - Immediate
- FB : Checksum (calculated for the string "0302")

Console Response:
Input is HIGH

Scheduled

Command: 0x03

Description: This command sets the output state of GPIO pin PA8 on Device 03 to a specified level (HIGH or LOW) after a user-defined delay. Upon execution, it first ensures PA8 is configured for output mode with no pull-up or pull-down resistors. An initial message "device3 scheduled a task on" is posted to the console. After the specified delay period has elapsed, PA8 is driven to the requested logic state (HIGH or LOW) based on the provided state parameter.

Parameters:
- Delay (16-bit): An unsigned integer representing the duration to wait before setting the PA8 output state.
- State (16-bit): An unsigned integer specifying the desired output state for PA8 after the delay.

Payload Structure:
- 03: Command Code for Set PA8 Output State with Delay.

- DH DL: Two hexadecimal bytes (DH = Delay High Byte, DL = Delay Low Byte) representing the 16-bit delay.
- SH SL: Two hexadecimal bytes (SH = State High Byte, SL = State Low Byte) representing the 16-bit desired state (0001 for HIGH, 0000 for LOW).

Confirmation Message:
Upon receiving the command, the device posts "device3 scheduled a task on" to the console. There is no further console message from this specific command routine after the pin state is changed.

MoT Command String Example:
:030300000001F9
- : : Start character
- 03 : Device 03 ID
- 03 : Command Code for Set PA8 Output State with Delay
- 0000 : Delay value (e.g., 0ms)
- 0001 : State value (HIGH)
- F9 : Checksum (calculated for the string "030300000001")

Console Response:
device3 scheduled a task on

Repetitive

Command: 0x04

Description:
This command generates a repetitive square wave output on GPIO pin PA8 of Device 03 for a specified number of cycles. Upon execution, it first ensures PA8 is configured for output mode with no pull-up or pull-down resistors. The command then enters a loop for the specified number of cycles. In each cycle: PA8 is held in a LOW state for the duration specified by the first time parameter. PA8 is then set HIGH. PA8 is held in a HIGH state for the duration specified by the second time parameter. PA8 is then set LOW. This sequence repeats for the defined number of cycles. An oscilloscope may be useful for verifying precise timings.

Parameters:
- Cycles (16-bit): An unsigned integer specifying the number of square wave cycles to generate.
- First Time Parameter (labeled TimeON in test script) (16-bit): An unsigned integer representing the duration (e.g., in milliseconds, as per test script examples) for which PA8 will be held LOW during each cycle before transitioning to HIGH.
- Second Time Parameter (labeled TimeOFF in test script) (16-bit): An unsigned integer representing the duration (e.g., in milliseconds, as per test script examples) for which PA8 will be held HIGH during each cycle before transitioning to LOW.

Payload Structure:
- 04: Command Code for Set PA8 Repetitive Output.
- CH CL: Two hexadecimal bytes (CH = Cycles High Byte, CL = Cycles Low Byte) representing the 16-bit number of cycles.
- P1H P1L: Two hexadecimal bytes (P1H = First Time Parameter High Byte, P1L = First Time Parameter Low Byte) representing the 16-bit duration PA8 is LOW.

- P2H P2L: Two hexadecimal bytes (P2H = Second Time Parameter High Byte, P2L = Second Time Parameter Low Byte) representing the 16-bit duration PA8 is HIGH.

Confirmation Message:
Upon receiving the command, the device posts "device3 repetitive mode is on" to the console.

MoT Command String Example:
:0304050007D007D046

- : : Start character
- 03 : Device 03 ID
- 04 : Command Code for Set PA8 Repetitive Output
- 0500 : Cycles value (5 cycles)
- 07D0 : First Time Parameter value (LOW = 2000ms)
- 07D0 : Second Time Parameter value (HIGH = 2000ms)
- 46 : Checksum (calculated for the string "0304050007D007D0")

Console Response:
device3 repetitive mode is on

Device 4: Digital-to-Analog Converter (DAC)

Introduction

Device 4 provides control over the STM32G491RE's Digital-to-Analog Converter (DAC), specifically DAC1 Channel 2, which outputs an analog voltage on pin PA5. This allows for setting constant DC voltages or generating simple periodic waveforms. The device also supports enabling an external trigger on pin PA1 for potential future use or specific triggered events.

Commands are sent to Device 4 as part of a MoT message. The first byte of the payload for Device 4 should contain the command code in its lower 4 bits (0-3).

Input Commands

Initialize DAC

Command: 0x00

Description: This command initializes the DAC1 Channel 2 peripheral and configures the associated GPIO pin (PA5) for analog output. It must be called before any other Device 4 output commands can be successfully executed.

Parameters: None.

Payload Structure:

- 00: Command Code for Initialize DAC.

Confirmation Message: Upon successful initialization, the device will post the message: "device has been initialized" to the console.

MoT Command String Example:
:0400FC

- : : Start character
- 04 : Device 4 ID
- 00 : Command Code for Initialize Device
- FC : Checksum of 0400

Console Response:

Enable External Trigger

Command: 0x04

Description: This command configures an external interrupt on pin PA1 (rising edge) and pre-loads a specific voltage value. When the PA1 pin experiences a rising edge, the

pre-loaded voltage will be output on DAC1 Channel 2 (PA5). This command sets an internal `trigger_state` to 1, indicating that Device 4 should handle the EXTI1 interrupt.

Parameters:

- Voltage (12-bit): An unsigned integer representing the DAC output value (0 to 4095 or 0x0FFF) to be applied when the trigger occurs. This value is sent as three or four ASCII hexadecimal characters (e.g., `FFF` or `0FFF`).

Payload Structure:

- 04 : Command Code
- 0VVV : A single zero padded 12 bit hex evaluate representing the desired output voltage

Confirmation Message: Upon successful execution of the command (trigger enabled), the device will post the message: "device4 Trigger Enabled on PA1" to the console.

MoT Command String Example (to output 0xFFF on trigger):
:04040 FFEA` (Checksum for "04040FFF" is EA)

- : : Start character
- 04 : Device 4 ID
- 04 : Command Code for Enable
- 0FFF : Voltage value (0xFFF) to
- EA : Checksum

Console Response:

Output Commands

Set Constant Voltage

Command: 0x01

Descrition: This command sets the DAC1 Channel 2 output to a specific, constant DC voltage.

Parameters:

- Voltage (12-bit): An unsigned integer representing the desired DAC output value (0 to 4095 or 0x0FFF). This value is sent as three or four ASCII hexadecimal characters (e.g., `7FF` or `07FF`).

Payload Structure:

- 01 : Command Code
- 0VVV : A single zero padded 12 bit hex evaluate representing the desired output voltage

Confirmation Message: Upon successful execution of the command (trigger enabled), the device will post the message: "device4 Trigger Enabled on PA1" to the console.

MoT Command String Example (to output 0xFFF on trigger):

:04040 FFEA

- : : Start character
- 04 : Device 4 ID
- 01 : Command Code for Enable
- 0FFF : Voltage value (0xFFF) to
- ED : Checksum

Console Response:

Start Waveform Output

Command: 0x02

Descrition: This command configures the DAC1 Channel 2 to output a periodic square waveform. The waveform alternates between two specified voltage levels (Va and Vb) with specified durations (T1 for Va and T2 for Vb) for each level. The waveform generation begins with outputting voltage Va for duration T1, then switches to Vb for duration T2, and this cycle repeats until explicitly stopped.

Parameters:

- Voltage A (Va) (12-bit, sent as 16-bit): An unsigned integer (0x000 to 0xFFF) representing the first DAC output voltage. Sent as 4 ASCII hexadecimal characters (e.g., 0FFF).
- Time 1 (T1) (16-bit): An unsigned integer representing the duration in milliseconds for Voltage A to be active. Sent as 4 ASCII hexadecimal characters (e.g., FFFF).
- Voltage B (Vb) (12-bit, sent as 16-bit): An unsigned integer (0x000 to 0xFFF) representing the second DAC output voltage. Sent as 4 ASCII hexadecimal characters (e.g., 0000).
- Time 2 (T2) (16-bit): An unsigned integer representing the duration in milliseconds for Voltage B to be active. Sent as 4 ASCII hexadecimal characters (e.g., FFFF).

Payload Structure:

- 02: Command Code for Start Waveform Output.
- VaVa: Voltage A, 4 hex characters (e.g., 0FFF).
- T1T1: Time 1, 4 hex characters (e.g., FFFF).
- VbVb: Voltage B, 4 hex characters (e.g., 0000).
- T2T2: Time 2, 4 hex characters (e.g., FFFF).

Confirmation Message: Upon successful execution, the device will post the message: "Voltage transitioned" to the console.

Confirmation Message:

MoT Command String Example (to generate a waveform with Va=0xFFF for T1=0xFFFF ms, and Vb=0x000 for T2=0xFFFF ms):
:04020FFFFFF0000FFFFF0

- : : Start character
- 04: Device 4 ID
- 02: Command Code for Start Waveform Output
- 0FFF: Voltage A (Va = 0xFFF)
- FFFF: Time 1 (T1 = 0xFFFF)

- 0000: Voltage B (Vb = 0x000)
- FFFF: Time 2 (T2 = 0xFFFF)
- F0: Checksum

Console Response:
Voltage transitioned

Stop Waveform Output

Command: 0x03

Descrition: This command stops any ongoing periodic waveform generation initiated by the "Start Waveform Output" command. It cancels the scheduled waveform tasks, sets the DAC1 Channel 2 output to zero (0V), and posts a confirmation message.

Parameters: None.

Payload Structure:
- 03: Command Code for Stop Waveform Output

Confirmation Message: Upon successful execution, the device will post the message: "device4 has stopped outputting a periodic signal" to the console.

MoT Command String Example:
:0403F9
- `:` : Start character
- `04` : Device 4 ID
- `03` : Command Code for Stop Waveform Output
- `F9` : Checksum (for "0403")

Console Response:
device4 has stopped outputting a periodic signal

Device 5: Analog-to-Digital Converter (ADC)

Introduction

Device 5 provides access to the STM32G491RE's Analog-to-Digital Converter (ADC). This allows the microcontroller to read analog voltage levels, (from PA0), and convert them into digital values. Device 5 can be used to report these digital values directly, compare them against thresholds, or potentially use external triggers for ADC operations. Commands are sent to Device 5 as part of a MoT message.

Input Commands

Initialize ADC

Command: 0x00

Description: This command initializes the ADC peripheral and configures the associated GPIO pin (e.g., PA0) for analog input. This command should be called before attempting to read ADC values.

Parameters: None

Payload Structure:

- 00: Command Code for Initialize ADC.

Confirmation Message: Upon successful initialization, the device will post: "Device 5 ADC initialized" to the console.

MoT Command String Example:
:0500 FB

- :: Start character
- 05: Device 5 ID
- 00: Command Code for Initialize ADC
- FB: Checksum (for "0500")

Console Response:
Device 5 ADC initialized

Report ADC Value

Command: 0x01

Description: This command reads the current digital value from the configured ADC channel (e.g., PA0) and posts it to the console. The value is a 12-bit number (0x000 to 0xFFF).

Parameters: None

Payload Structure:

- 01: Command Code for Report ADC Value.

Confirmation Message: The console will display the current ADC reading, for example: "Device 5 ADC Value: 0xXXX"

MoT Command String Example:
:0501FA
- :: Start character
- 05: Device 5 ID
- 01: Command Code for Report ADC Value
- FA: Checksum (for "0501")

Console Response:
Device 5 ADC Value: 0x7FF

Report if ADC Value is Above/Below Thresholds

Command: 0x02

Description: This command reads the ADC value and compares it against a specified lower and upper threshold. It then posts a message to the console indicating if the current voltage is below the lower threshold, above the upper threshold, or within the specified range.

Parameters:
- Lower Threshold (12-bit, sent as 16-bit): An unsigned integer (0x000 to 0xFFF) representing the lower limit. Sent as 4 ASCII hexadecimal characters (e.g., 0100).
- Upper Threshold (12-bit, sent as 16-bit): An unsigned integer (0x000 to 0xFFF) representing the upper limit. Sent as 4 ASCII hexadecimal characters (e.g., 0FAA).

Payload Structure:
- 02: Command Code.
- LLLL: Lower Threshold, 4 hex characters (e.g., 0100).
- HHHH: Upper Threshold, 4 hex characters (e.g., 0FAA).

Confirmation Message: Depending on the ADC value relative to the thresholds, messages like "Device 5: Voltage BELOW LIMIT", "Device 5: Voltage ABOVE LIMIT", or "Device 5: Voltage WITHIN LIMITS" will be posted.

MoT Command String Example:
:050201000FAA3F
- : : Start character
- 05 : Device 5 ID
- 02 : Command Code
- 0100 : Lower Threshold (0x100)
- 0FAA : Upper Threshold (0xFAA)
- 3F : Checksum

Console Response:
Device 5: Voltage BELOW LIMIT

Stop Monitor Task

Command: 0x03

Description: This command stops any active monitoring task associated with Device 5, such as continuous threshold checking or periodic reporting.

Parameters: None

Payload Structure:

- 03: Command Code for Stop Monitor.

Confirmation Message: A message confirming the action, for example: "Device 5 monitoring stopped".

MoT Command String Example:
:0503F8

- :: Start character
- 05: Device 5 ID
- 03: Command Code for Stop Monitor
- F8: Checksum (for "0503")

Console Response:
Device 5 monitoring stopped

Initialize External Trigger (EXTI1)

Command: 0x04

Description: This command initializes and enables an external interrupt on pin PA1. The specific action triggered by this interrupt for Device 5 (e.g., starting an ADC conversion, reporting a value) depends on the firmware implementation.

Parameters: None

Payload Structure:

- 04: Command Code for EXTI1 Init.

Confirmation Message: A message indicating the trigger has been set, for example: "Device 5 EXTI1 initialized on PA1"

MoT Command String Example:
:0504F7

- : : Start character
- 05 : Device 5 ID
- 04 : Command Code for EXTI1 Init
- F7 : Checksum (for "0504")

Console Response:

Device 5 EXTI1 initialized on PA1

Device 6: TIMER (TIM2)

Introduction

Device 6 provides control over the TIM2 timer peripheral on the STM32G491RE, with output functionalities directed to pin PA5. This device allows for the generation of various time-based signals, including Pulse Width Modulation (PWM), Pulse Frequency Modulation (PFM), single pulses, one-pulse mode (OPM) outputs, and pulse trains at specified frequencies. It also includes functionality for trigger initialization and a general command to turn off active timer functions, which is useful for sequencing different timer operations. Care should be taken if Device 4 (DAC1 Channel 2 on PA5) is also intended to be used, as they share the same output pin.

Output Commands

PWM On (Pulse Width Modulation)

Command: 0x01

Description: This command configures and enables Pulse Width Modulation (PWM) on pin PA5, which is connected to TIM2 Channel 1. The command sets the PWM period using the Auto-Reload Register (ARR) value and the pulse width (active high time) based on the provided duty cycle percentage. The PWM signal is generated continuously until the timer function is turned off or reconfigured.

Parameters:

- ARR (Auto-Reload Register Value) (4-byte): An unsigned 32-bit integer that defines the period of the PWM signal.
- Duty Cycle Percentage (2-byte): An unsigned 16-bit integer representing the desired duty cycle as a percentage.

Payload Structure:

- 01: Command Code for PWM On.
- AAAAAAAA: ARR Value, 8 hexadecimal characters representing the 32-bit value.
- DDDD: Duty Cycle Percentage, 4 hexadecimal characters representing the 16-bit value (0-100).

Confirmation Message:
Upon successful execution, the device will post the message: "device6 PWM ON" to the console.

MoT Command String Example:
:0601001E847F0019BF

- :: Start character
- 06: Device 6 ID
- 01: Command Code for PWM On
- 001E847F: ARR value (1,999,999). With a 1MHz timer clock, this results in a period of (1999999+1) * 1µs = 2 seconds.

- 0019: Duty Cycle Percentage (25 decimal, for 25% duty cycle). The active pulse width will be 25% of 2 seconds = 0.5 seconds.
- BF: Checksum.

Console Response:
device6 PWM ON

PFM On (Pulse Frequency Modulation)

Command: 0x02

Description: This command configures TIM2 Channel 1 (outputting on pin PA5) to generate a continuous square wave signal at a user-specified target frequency. The duty cycle of this square wave is fixed at 50%.

Parameters:
- Target Frequency (4-byte): An unsigned 32-bit integer representing the desired output frequency in Hertz (Hz).

Payload Structure:
- 02: Command Code for PFM On.
- FFFFFFFF: Target Frequency (Hz), 8 hexadecimal characters representing the 32-bit value.

Confirmation Message: Upon successful execution, the device will post the message: "device6 PFM ON" to the console.

MoT Command String Example:
- :: Start character
- 06: Device 6 ID
- 02: Command Code for PFM On
- 00000064: Target Frequency value (100 Hz).
- 94: Checksum.

Console Response:
device6 PFM ON

Simple Pulse On

Command: 0x03

Description: This command generates a single digital pulse of a specified width on pin PA5 (TIM2 Channel 1). It utilizes TIM2's One-Pulse Mode (OPM).

Parameters:
- Pulse Width (4-byte): An unsigned 32-bit integer representing the desired duration of the pulse in timer ticks.

Payload Structure:
- 03: Command Code for Simple Pulse On.
- WWWWWWWW: Pulse Width, 8 hexadecimal characters representing the 32-bit value in timer ticks.

Confirmation Message: Upon successful execution, the device will post the message: "device6 Pulse ON" to the console.

MoT Command String Example:
:0603000F423F67

- : : Start character
- 06 : Device 6 ID
- 03 : Command Code for Simple Pulse On
- 000F423F : Pulse Width in timer ticks (999,999 decimal). With a 1MHz timer clock, this is a pulse of 999,999 µs.
- 67 : Checksum.

Console Response:
device6 Pulse ON

Pulse Width On

Command: 0x04

Description: This command configures TIM2 Channel 1 (outputting on pin PA5) to generate a single pulse with a user-specified delay and a user-specified pulse width. The function utilizes the timer's One-Pulse Mode (OPM).

Parameters:

- Delay (4-byte): An unsigned 32-bit integer representing the delay before the pulse starts, in timer ticks.
- Pulse Width (4-byte): An unsigned 32-bit integer representing the duration of the active pulse, in timer ticks.

Payload Structure:

- 04: Command Code for One-Pulse Mode (OPM) On.
- DDDDDDDD: Delay in timer ticks, 8 hexadecimal characters representing the 32-bit value.
- WWWWWWWW: Pulse Width in timer ticks, 8 hexadecimal characters representing the 32-bit value.

Confirmation Message: Upon successful execution, the device will post the message: "device6 Pulse Width ON" to the console.

MoT Command String Example:
:0604001E847F001E847FB4

- : : Start character
- 06 : Device 6 ID
- 04 : Command Code
- 001E847F : Delay value in timer ticks (1,999,999 decimal). With a 1MHz timer clock, this is a delay of 1,999,999 µs.
- 001E847F : Pulse Width value in timer ticks (1,999,999 decimal). With a 1MHz timer clock, this is a pulse duration of 1,999,999 µs.
- B4 : Checksum.

Console Response:

device6 Pulse Width ON

Pulse Frequency On

Command: 0x05

Description: Generates a single pulse at a specified target frequency. The duty cycle of these pulses is fixed at 50%.

Parameters:

- Target Hz (32-bit): An unsigned integer representing the desired pulse frequency in Hertz. Sent as 8 ASCII hexadecimal characters (e.g., 00000001 for 1 Hz).

Payload Structure:

- 05: Command Code.
- FFFF FFFF: Target Frequency (Hz), 8 hex characters.

Confirmation Message: A message indicating pulse train is active, for example: "Device 6 Pulse Frequency active: Target=X Hz"

MoT Command String Example (for Target Frequency = 1 Hz):
:060500000001F4

- : : Start character
- 06 : Device 6 ID
- 05 : Command Code
- 00000001 : Target Frequency value (1 Hz)
- F4 : Checksum

Console Response:
Device 6 Pulse Frequency active

Initialize Trigger

Command: 0x06

Description: Initializes or enables a trigger mechanism for the timer. The command prepares the timer to respond to a trigger event on PA1. Once PA1 is triggered one of the input functions of the timer will execute depending on what was the input of the triggered control.

Parameters: None.

Payload Structure:

- 06: Command Code for Initialize Trigger.

Confirmation Message: A message confirming trigger initialization, for example: "Device 6 Trigger Initialized".

MoT Command String Example:
:0606F4

- : : Start character
- 06 : Device 6 ID

- 06 : Command Code
- F4 : Checksum (for "0606")

Console Response:
Device 6 Trigger Initialized

Triggered Control

Command: 0x07

Description: This command arms a specific timer output function to be executed when an external trigger event occurs on pin PA1. The trigger input on PA1 must be initialized separately using the 'Initialize Trigger' (Command `0x06`) command before this command can be effectively used. The 'Triggered Control' command requires a sub-command to select the desired timer function (e.g., PWM, PFM, Simple Pulse, Pulse Width, or Pulse Frequency) along with its specific parameters. These parameters are stored internally, and the selected timer operation will commence upon receiving a trigger signal on PA1. Only one timer function can be armed to respond to the trigger at any given time.

Parameters:

- 07: Command Code for Tiggered Control.
- Sub-Command (1 byte): A single hexadecimal byte specifying the timer function to arm. The supported sub-commands and their subsequent parameters are:
    - 01 (PWM - Pulse Width Modulation)
        - ARR (4 bytes): Auto-Reload Register value, determining the PWM period.
        - Duty (2 bytes): Capture/Compare Register value, determining the PWM pulse width (duty cycle).
    - 02 (PFM - Pulse Frequency Modulation):
        - Target Hz (4 bytes): The desired frequency in Hertz. The firmware calculates the ARR for this frequency and uses a fixed 50% duty cycle.
    - 03 (SP - Simple Pulse)
        - Pulse Width (4 bytes): Duration of the single pulse in timer ticks.
    - 04 (PW - Pulse Width)
        - Delay (4 bytes): Delay before the pulse starts, in timer ticks.
        - Width (4 bytes): Duration of the pulse, in timer ticks.
    - 05 (PF - Pulse Frequency On / Single Cycle Pulse):
        - Target Hz (4 bytes): The desired frequency in Hertz for a single 50% duty cycle pulse.

Payload Structure:
:0607<SubCmd><Parameters...><CS>

- <SubCmd>: The 1-byte sub-command.
- <Parameters...>: Variable length data bytes specific to the chosen sub-command. For example:
    - PWM (01): 01<ARR_4B><Duty_2B>
    - PFM (02): 02<TargetHz_4B>
    - SP (03): 03<PulseWidth_4B><OptionalExtraByte_1B> (as per demo example)
    - OPM (04): 04<Delay_4B><Width_4B>
    - PF (05): 05<TargetHz_4B>

Confirmation Message: Upon successfully arming the selected function, the device will post the message: "device6 Trigger Armed With Given Command\n\r" to the console.

MoT Command String Example:
The last byte in all of the commands below is a Check Sum
:060701001E847F0019B8

- To arm PWM with ARR = 0x001E847F and Duty = 0x0019:

:06070200000001F0

- To arm PFM for 1 Hz:

:060703001E847FCF

- To arm Simple Pulse with Width = 0x001E847F

:060704001E847F001E847FB4

- To arm One-Pulse Mode with Delay = 0x001E847F and Width = 0x001E847F.

:06070500000001ED

- To arm Pulse Frequency for 1 Hz.

Console Response:
device6 Trigger Armed With Given Command

Timer Function Off

Command: 0x08

Description: Disables the currently active timer function (e.g., PWM, PFM, Pulse train). This command is useful for stopping a timer operation before configuring another or to ensure the timer output is inactive.

Parameters: None.

Payload Structure:

- 08: Command Code for Timer Function Off.

Confirmation Message: A message confirming the timer function is off, for example: "Device 6 Timer Function Off"

MoT Command String Example:
:0608F2

- :: Start character
- 06: Device 6 ID
- 08: Command Code
- F2: Checksum (for "060B")

Console Response:
Device 6 Timer Function Off

Device 7: SPI Loopback Testing

Introduction

Input Commands

This section of the manual describes Device 7, which facilitates Serial Peripheral Interface (SPI) communication, specifically configured for loopback testing. In a loopback test, the Master Out Slave In (MOSI) pin is connected to the Master In Slave Out (MISO) pin, allowing the microcontroller to send data and receive the same data back, verifying the SPI communication path.

Initialize SPI

Command: 0x00

Description: Initializes the SPI peripheral for loopback testing, which involves enabling the clocks for the necessary GPIO port and the SPI peripheral itself, including performing a reset of the SPI peripheral. This command also configures the relevant GPIO pins for their roles in SPI communication (such as Chip Select, Clock, MISO, and MOSI) and sets up the SPI peripheral's operational parameters like master mode, data size, baud rate, clock polarity/phase, and software slave management, before finally enabling the SPI communication. It is necessary to call this command to prepare the hardware before any other Device 7 SPI operations can be successfully executed

Parameters: None.

Payload Structure:

- `00` : Command Code for Initialize SPI.

Confirmation Message: Upon successful initialization, the device will post the message: "device7: SPI2 initialized (PB12-15)" to the console.

MoT Command String Example:
:0700F9

- : : Start character
- 07 : Device 07 ID
- 00 : Command Code for Initialize SPI
- F9 : Checksum (calculated for the string "0700")

Console Response:
device7: SPI2 initialized (PB12-15)

SPI Write

Command: 0x01

Description: This command writes a specified number of data bytes (up to 10) to the SPI2 bus and simultaneously captures the data looped back on MISO into an internal receive buffer. It's recommended to clear the buffers using the 'Clear SPI Buffers' (Command 0x03) command before issuing this write command.

Parameters:

- Count (8-bit): A single hexadecimal byte specifying the number of data bytes to write (N). This count should not exceed the internal buffer capacity (10 bytes).
- Data Bytes (N bytes): The N data bytes to be written, each as a two-digit hexadecimal number.

Payload Structure:

- 01 : Command Code for Write Data and Capture Loopback.
- XX : Count of data bytes (1 byte).
- YY… : Data bytes (N bytes).

Confirmation Message:
Upon successful completion of the write and capture operations, the device will post the message: "device7: SPI2 write complete" to the console.

MoT Command String Example (to write 10 bytes: AA, BB, CC, DD, EE, FF, 11, 22, 33, 44):
:07010AAABBCCDDEEFF1122334449

- : : Start character
- 07 : Device 07 ID
- 01 : Command Code for Write Data and Capture Loopback
- 0A : Count (10 bytes)
- AABBCCDDEEFF11223344 : Data Bytes
- 49 : Checksum

Console Response:
device7: SPI2 write complete

Clear SPI Buffers

Command: 0x03

Description: This command clears the internal SPI read and write buffers, ensuring that any data from previous operations is removed and buffer access pointers are reset. This command is useful to call before a new sequence of SPI write and read operations to ensure a clean state and prevent data from prior transactions from affecting new ones.

Parameters: None.

Payload Structure:

- 03: Command Code for Clear SPI Buffers.

Confirmation Message: Upon successful execution, the device will post the message: "device7: Read and Write Buffers CLEAR" to the console.

MoT Command String Example:
:0703F6

- : : Start character
- 07 : Device 07 ID
- 03 : Command Code for Clear SPI Buffers
- F6 : Checksum (calculated for the string "0703")

Console Response (Example if the buffer contained AA, BB, ..., 44):
device7: 0xAABBCCDDEEFF11223344

Output Commands

SPI Read

Command: 0x02

Description: This command reads up to 10 bytes of data from an internal buffer, which should contain data captured during previous SPI write operations (via loopback), and reports this data to the console as a string of hexadecimal characters.

Parameters: None.

Payload Structure:
- 02 : Command Code for SPI Read.

Confirmation Message: The console will display the current content of the SPI read buffer (up to 10 bytes) in hexadecimal format, for example: "device7: 0xAABBCCDDEEFF11223344" (the actual hexadecimal values depend on the data captured in the buffer)

MoT Command String Example:
:07020AED
- : : Start character
- 07 : Device 07 ID
- 02 : Command Code for SPI Read
- 0A : Optional data byte (ignored by firmware)
- ED : Checksum

Console Response (Example if the buffer contained AA, BB, ..., 44):
device7: 0xAABBCCDDEEFF11223344

Device 8: W25QXX Flash Memory Storage

Introduction

This section details the functionality and usage of Device 8, which provides an interface to an external W25QXX series SPI Flash memory chip. You will learn how to perform fundamental operations such as initializing the flash memory, reading its JEDEC identification, erasing sectors, writing data to the flash, verifying the written data, and reading data back from the flash memory. These capabilities allow for non-volatile storage that can be accessed and managed through the MoT system.

Control Commands

Reset/Initialization

Command: 0x00

Description: This command initializes the W25QXX flash memory chip and the SPI peripheral (SPI2) used to communicate with it.

Parameters: None.

Payload Structure:

- 02 : Command Code for reset/initalization of FLASH.

Confirmation Message: Upon successful execution, the device will post the message: "device8 has been initalized on W25Q128" to the console.

MoT Command String Example:
:0800F8

- : : Start character
- 08 : Device ID for W25QXX Flash Memory.
- 00 : Command ID for Reset/Initialization.
- F8 : Checksum.

Console Response:
device8 has been initalized on W25Q128

Erase

Command: 0x02

Description: This command erases a specific sector (Sector 0, starting at address 0x000000, 4KB in size) in the W25QXX flash memory. This operation prepares the sector for new data to be written. It is important to erase a sector before attempting to write new data to it, as flash memory bits can only be changed from 1 to 0 during a write, and an erase operation resets them to 1. The address of the sector to be erased is currently fixed within the firmware to 0x000000.

Parameters: None.

Payload Structure:
- 02 : Command Code for Erase Sector.

Confirmation Message: Upon successful execution, the device will post the message: "device8 erased directed block" to the console.

MoT Command String Example:
:0802F6
- : : Start character
- 08 : Device ID for W25QXX Flash Memory.
- 02 : Command ID for Erase Sector.
- F6 : Checksum (calculated for the string "0802")

Console Response:
device8 erased directed block

Copy RAM Buffer to Flash

Command: 0x04

Description: This command writes the content of the internal RAM buffer to the W25QXX flash memory. The firmware is currently configured to write this data starting at flash address 0x000000. Ensure the target sector in flash memory has been erased (using Command 0x02) before executing this command.

Parameters: None.

Payload Structure:
- 04 : Command Code for Copy RAM Buffer to Flash.

Confirmation Message: Upon successful execution, the device will post the message: "device8 copy complete" to the console.

MoT Command String Example:
:0804F4
- : : Start character
- 08 : Device ID for W25QXX Flash Memory.
- 04 : Command ID for Copy RAM Buffer to Flash.
- F4 : Checksum (calculated for the string "0804")

Console Response:
device8 copy complete

Verify Flash against RAM Buffer

Command: 0x05

Description: This command compares the content of the W25QXX flash memory (starting at address 0x000000) with the content of the internal RAM buffer. It reads data from the flash

into a separate receive buffer and then compares this receive buffer byte-by-byte against the RAM buffer that was intended to be written. This is used to verify that the data was written to the flash correctly.

Parameters: None.

Payload Structure:
- 05 : Command Code for Verify Flash against RAM Buffer.

Confirmation Message: Upon successful execution, the device will post: "device8 verification successful" if the contents match. "device8 verification failed" if there is a mismatch.

MoT Command String Example:
:0805F3
- : : Start character
- 08 : Device ID for W25QXX Flash Memory.
- 05 : Command ID for Verify Flash against RAM Buffer.
- F3 : Checksum (calculated for the string "0805")

Console Response:
device8 verification successful
device8 verification failed

Input Commands

Input Data to RAM Buffer

Command: 0x03

Description: This command loads specified data bytes from the MoT command string into an internal RAM buffer on the microcontroller. This RAM buffer is then used as the source for writing data to the flash memory. The first byte of the data payload specifies the count of data bytes that follow.

Parameters:
- Count (8-bit): A single hexadecimal byte specifying the number of data bytes to load into the RAM buffer.
- Data Bytes (N bytes): The N data bytes to be loaded, each as a two-digit hexadecimal number.

Payload Structure:
- 03 : Command Code for Input Data to RAM Buffer.
- XX : Count of data bytes (1 byte).
- YY… : Data bytes (N bytes, as specified by XX).

Confirmation Message: Upon successful execution, the device will post the message: "device8 input is in ram" to the console.

MoT Command String Example (loading 8 bytes: AABBCCDDEEFF1122):
:080308AABBCCDDEEFF1122BF

- : : Start character
- 08 : Device ID for W25QXX Flash Memory.
- 03 : Command ID for Input Data to RAM Buffer.
- 08 : Count of data bytes (8).
- AABBCCDDEEFF1122 : Data bytes.
- BF : Checksum (calculated for the string "080308AABBCCDDEEFF1122")

Console Response:
device8 input is in ram

Output Commands

Read JDEC

Command: 0x01

Description: This command reads the JEDEC (Joint Electron Device Engineering Council) identification data from the W25QXX flash memory chip. This data includes the Manufacturer ID, Device Type ID (Memory Type), and Capacity ID. This is useful for verifying the correct flash chip is connected and operational.

Parameters: None.

Payload Structure:
- 01 : Command Code for reading the JDEC

Confirmation Message: Upon a successful read the following message will be displayed in the console: “device8 JEDEC: MF=0xXX, Type=0xYY, Cap=0xZZ”
- Where `XX` is the Manufacturer ID (e.g., EF for Winbond).
- Where `YY` is the Device Type ID (e.g., 40 for W25Q128FV Memory Type).
- Where `ZZ` is the Capacity ID (e.g., 18 for 128M-bit).

MoT Command String Example:
:0801F7
- : : Start character.
- 08 : Device ID for W25QXX Flash Memory.
- 01 : Command ID for Read JEDEC ID.
- F7 : Checksum.

Console Response:
device8 JEDEC: MF=0xEF, Type=0x40, Cap=0x18

Output Data from Flash

Command: 0x06

Description: This command reads a specified number of bytes from the W25QXX flash memory (starting at address 0x000000) and outputs them to the console as a hexadecimal string. The first byte of the data payload specifies the number of bytes to read and output.

Parameters:

- Count (8-bit): A single hexadecimal byte specifying the number of data bytes to read from flash and output.

Payload Structure:

- 06 : Command Code for Output Data from Flash.
- XX : Count of data bytes to read and output (1 byte).

Confirmation Message: The device will post a message containing the data read from flash, formatted as a hexadecimal string. For example: "device8 output from flash 0xAABBCCDDEEFF1122..." (the rest of the buffer might be shown as zeros if fewer than 32 bytes are requested, depending on the fixed message buffer size).

MoT Command String Example (to output 8 bytes):
:080608EA

- : : Start character
- 08 : Device ID for W25QXX Flash Memory.
- 06 : Command ID for Output Data from Flash.
- 08 : Number of bytes to read and output.
- EA : Checksum (calculated for the string "080608")

Console Response (Example for 8 bytes AABBCCDDEEFF1122):
device8 output from flash 0xAABBCCDDEEFF11220000000000000000000000000000000000

End of document