# Customer training workshop: EVTGEN_trigger_ADC for KIT_T2G-B-H_EVK

TRAVEO$^{TM}$ T2G CYT4BF series Microcontroller Training
V1.0.0 2022-11

Please read the **Important notice and warnings** at the end of this document

# Scope of work

- This code example demonstrates how to use the TRAVEO™ T2G MCU Event Generator (EVTGEN) resource to trigger ADC conversion in Active power mode. In this example, the Event Generator is configured to trigger an ADC conversion every second, and when the ADC conversion is complete, print out the ADC result via UART.

- Device
  - The TRAVEO™ T2G CYT4BFBCH device is used in this code example.

- Board
  - The TRAVEO™ T2G KIT_T2G-B-H_EVK board is used for testing.

# Introduction

- **Event Generator (EVTGEN) has the following features:**
  - CPU-free triggers for device functions
  - Reduces CPU involvement in triggering device functions; reducing overall power consumption and CPU bandwidth
  - 16 comparators for each DeepSleep and Active mode to generate interrupts and triggers
  - 32-bit counter, one each for DeepSleep and Active mode for comparators
  - Individual configurable thresholds for each comparator
  - DeepSleep and Active mode clock sources for counters
  - Jitter-free initiation of specific device functionality
  - One DeepSleep and one Active mode interrupt for CPU
  - Supported in Active, Sleep, LPActive, LPSleep, and DeepSleep power modes

# Introduction (contd.)

- **ADC has the following features:**
  - SAR ADC Core
    - 12-bit resolution with a maximum sample rate of 1 Msps
  - 32 logical channels with the same capabilities
  - Each logical channel can select input from
    - 32 analog input pins
    - Diagnostic signals
    - Analog input pins of other ADC units
    - Support for external mux (three select bits)
    - AMUXBUSA/B
  - Scans triggered by timer, software, continuous, pins, or system triggers
    - Multiple ADC units can be triggered by the same trigger to ensure lock-step operation
    - Triggers can be cleared by software
    - Optional debug pause
  - Double buffering of output data
  - Programmable sample time for each channel

# Introduction (contd.)

- **ADC has the following features:**
  - Programmable post processing options for each channel
    - Sign/zero extension to 16-bit
    - Left/right alignment
    - Averaging: first order accumulate and dump, up to 256 samples
    - Programmable right shift
    - Range detection: below/above threshold, in/out-side range
    - Pulse detection: programmable positive and negative event counters
  - Channels can be individual or grouped
    - Flexible grouping: from 32 groups with one channel to one group with 32 channels
  - Group scans are dynamically scheduled by the hardware
    - Eight priorities, programmable per group
    - Four preemption types: resume, restart, cancel, or finish
    - Optional automatic idle power down

# Introduction (contd.)

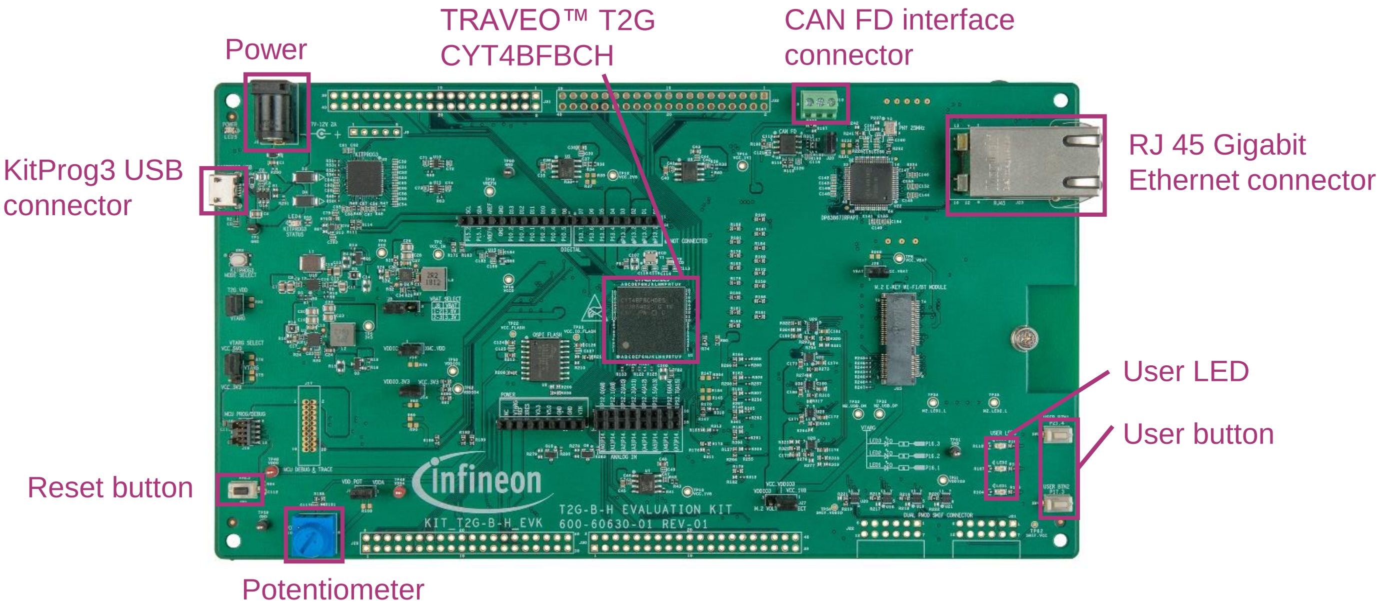
- **ADC has the following features:**
  - Interrupt generation
    - Group scan done
    - Group scan done overflow detect
    - Group scan canceled
    - Per channel range detect
    - Per channel pulse detect
    - Per channel pulse/range overflow detect
  - Output trigger generation per channel
    - Data ready/completion (each channel can trigger DW transfer)
    - Range violation detected
  - Digital and analog calibration available
  - Programmable offset and gain calibration
    - Non-intrusive background recalibration
    - Coherent calibration update

# Introduction (contd.)

- **ADC has the following features:**
  - Support for diagnostic measurements including broken wire detection. This includes:
    - ADC sampling capacitor preconditioning feature
    - Selectable current source or sink on selected ADC input while sampling
    - Support for LED diagnostics
  - On-chip temperature sensor and power monitoring

# Hardware setup

- This code example has been developed for the KIT-T2G-B-H-EVK board.
- Connect your PC to the board using the provided USB cable through the KitProg3 USB connector.



# Implementation

- This code example demonstrates how to use the Event Generator in Active power mode. The counter clock of the Event Generator block is configured to 1 MHz, and the active compare value is configured to 1000000. When the active counter is greater than or equal to the active compare value, it will generate the active trigger event to trigger SAR ADC conversion and interrupt. In the Event Generator interrupt handler, it will update the active comparator value and generate events every second to trigger ADC conversions. When SAR ADC conversion is complete, print out the ADC result via UART.

**Follow these steps to configure this code example:**

- STDOUT setting
- ADC initialization
- ADC interrupt configuration
- Event Generator interrupt configuration
- Event Generator initialization
- Start Event Generator
- Event Generator comparator structure initialization
- Update active comparator value
- Get ADC conversion result

# Implementation (contd.)

## STDOUT setting

- Call the ***cy_retarget_io_init()*** function to use UART as STDOUT.
  - Initializes P13.1 as UART TX, P13.0 as UART RX (these pins are connected to KitProg3 COM port)
  - The serial port parameters change to 8N1 and 115200 baud

## ADC initialization

- Call the ***CY_SAR2_Init()*** function to initialize the ADC channel.
  - Initialize the ADC channel 0 to use pin P6.6 as input

## ADC interrupt configuration

- Call the ***Cy_SAR2_Channel_SetInterruptMask()*** function to configure the channel interrupt.
  - Specify ***CY_SAR2_INT_GRP_DONE*** as an argument to generate an interrupt when the conversion is completed
- Configure interrupt in the ***CY_SysInt_Init()*** function.
  - Set the interrupt source (ADC channel 0), interrupt priority (7), interrupt vector, and the ISR (***adc_int_handler()***)
- Then, clear the IRQ request of the configured interrupt by ***NVIC_ClearPendingIRQ()***[1], before enabling IRQ by ***NVIC_EnableIRQ()***[1].

[1]: The CPU interrupt enable and NVIC operation instructions are provided by Cortex microcontroller software interface standard (CMSIS) with intrinsic functions.

# Implementation (contd.)

## Event Generator interrupt configuration

- Configure interrupt in the ***CY_SysInt_Init()*** function.
  - Set the interrupt source (EVGEN0), interrupt priority (7), interrupt vector, and the ISR (***evtgen_isr()***)
- Call the ***Cy_EvtGen_ClearInterrupt()*** function.
  - Set the EVTGEN0 bit to 0 to clear the interrupt
- Then, clear the IRQ request of the configured interrupt by ***NVIC_ClearPendingIRQ()***[1], before enabling IRQ by ***NVIC_EnableIRQ()***[1].

## Event Generator initialization

- Call the ***Cy_EvtGen_Init()*** function to initialize the Event Generator.
  - Initializes Event Generator parameters (clock source frequency in Active/DeepSleep power mode, EVTGEN customized period, ratio control mode and specific dynamic mode)
  - See also ***cy_stc_evtgen_config_t*** for parameter details

[1]: The CPU interrupt enable and NVIC operation instructions are provided by Cortex microcontroller software interface standard (CMSIS) with intrinsic functions.

# Implementation (contd.)

### Start Event Generator

- Call the ***Cy_EvtGen_Enable()*** function to start the Event Generator.
  - Enable the Event Generator
- Call the ***Cy_SysLib_DelayUs()*** function to set the delay bit and wait for the counter to complete initialization.
  - Waiting for 625 µsec
- Call the ***Cy_EvtGen_GetRatioStatus()*** function check to determine if the ratio status is valid during hardware control ratio.
  - Get VALID bit of EVTGEN0_RATIO_CTL register
- Call the ***Cy_EvtGen_GetCounterStatus()*** function to check if the Event Generator counter status is valid.
  - Get VALID bit of EVTGEN0_COUNTER_STATUS register

# Implementation (contd.)

### Event Generator comparator structure initialization

- Call the ***Cy_EvtGen_InitStruct()*** function to initialize the Event Generator comparator structure.
  - Initializes the Event Generator parameters (functionality comparator structure, condition for start trigger/interrupt, making period of interrupts/triggers and the making period of interrupts during DeepSleep)
  - Refer to ***cy_stc_evtgen_struct_config_t*** for parameter details

### Update active comparator value

- Check that the interrupt source is EVTGEN0 with the return value of the ***Cy_EvtGen_GetStructInterrupt()*** function.
  - Get interrupt flag of EVTGEN0
- Call the ***Cy_EvtGen_ClearStructInterrupt()*** function to clear the interrupt factor.
  - Clear interrupt flag of EVTGEN0
- Call the ***Cy_EvtGen_UpdateActiveCompValue()*** function to update the active comparator value.
  - Update active comparator to initial value (1000000 = 1sec); this can be modified to change the ADC conversion cycle

# Implementation (contd.)

## Get ADC conversion result

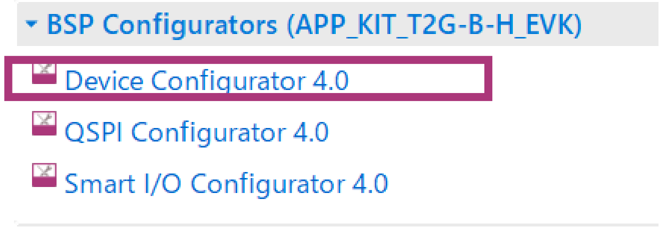
- Call the ***Cy_SAR2_Channel_GetResult()*** function to get the ADC conversion result and status.
- Call the ***Cy_SAR2_Channel_ClearInterrupt()*** function to clear the interrupt factor.
  - Clear interrupt flag of specified ADC channel (channel 0 of ADC0)

# Implementation (contd.)

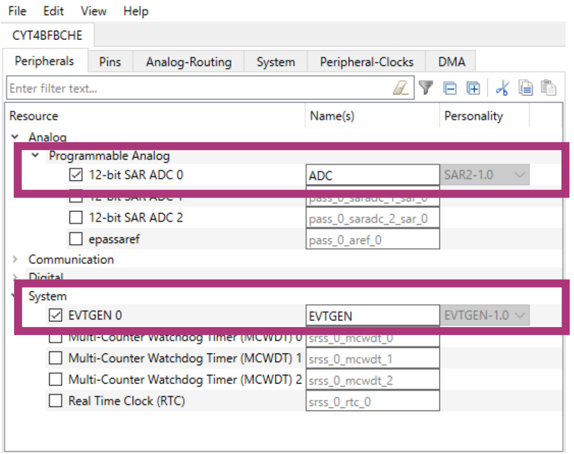
## Configure Event Generator and ADC

- You can change Event Generator and ADC configuration by Device Configurator.

1. Open Device Configurator.
2. Select "**Peripherals**" tab.
3. Select "**12-bit SAR ADC 0**" from Programmable Analog under Analog.
4. Select "**EVTGEN 0**" from System under Resource.



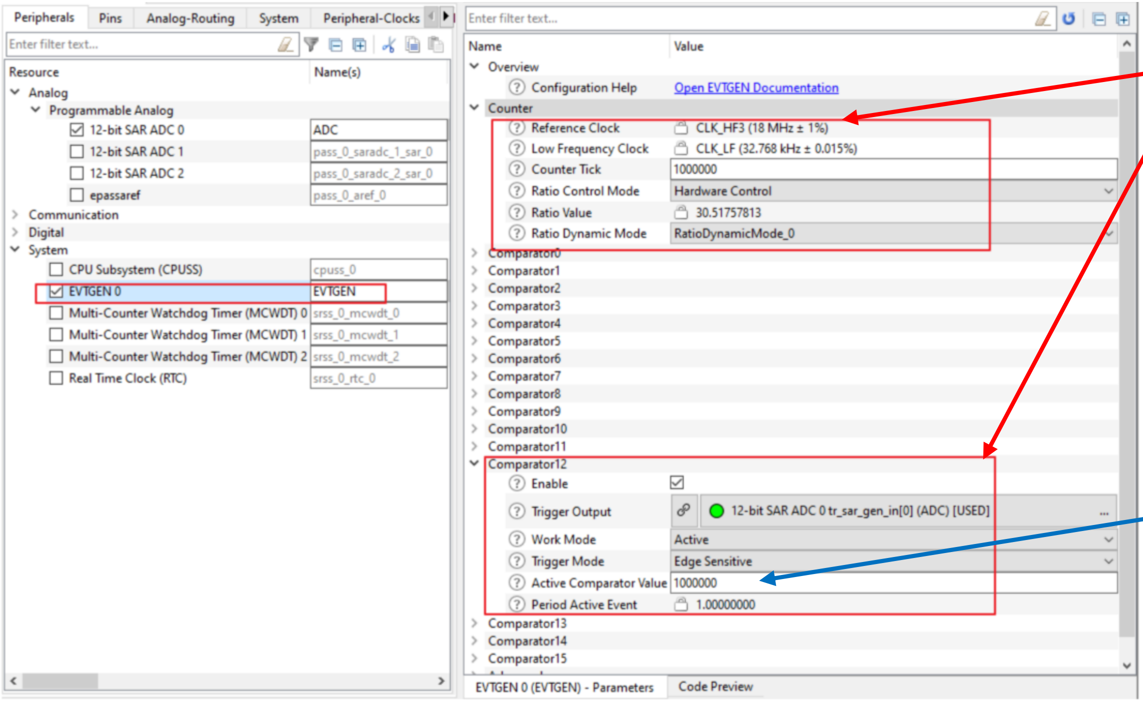
BSP Configurators (APP_KIT_T2G-B-H_EVK)
Device Configurator 4.0
QSPI Configurator 4.0
Smart I/O Configurator 4.0



| Resource | Name(s) | Personality |
|---|---|---|
| Analog | | |
| Programmable Analog | | |
| ☑ 12-bit SAR ADC 0 | ADC | SAR2-1.0 |
| □ 12-bit SAR ADC 1 | pass_0_saradc_1_sar_0 | |
| □ 12-bit SAR ADC 2 | pass_0_saradc_2_sar_0 | |
| □ epassaref | pass_0_aref_0 | |
| Communication | | |
| Digital | | |
| System | | |
| ☑ EVTGEN 0 | EVTGEN | EVTGEN-1.0 |
| □ Multi-Counter Watchdog Timer (MCWDT) 0 | srss_0_mcwdt_0 | |
| □ Multi-Counter Watchdog Timer (MCWDT) 1 | srss_0_mcwdt_1 | |
| □ Multi-Counter Watchdog Timer (MCWDT) 2 | srss_0_mcwdt_2 | |
| □ Real Time Clock (RTC) | srss_0_rtc_0 | |

# Implementation (contd.)

## Event Generator configuration



| Resource | Name(s) |
|---|---|
| Analog | |
| Programmable Analog | |
| ☑ 12-bit SAR ADC 0 | ADC |
| □ 12-bit SAR ADC 1 | pass_0_saradc_1_sar_0 |
| □ 12-bit SAR ADC 2 | pass_0_saradc_2_sar_0 |
| □ epassaref | pass_0_aref_0 |
| Communication | |
| Digital | |
| System | |
| □ CPU Subsystem (CPUSS) | cpuss_0 |
| ☑ EVTGEN 0 | EVTGEN |
| □ Multi-Counter Watchdog Timer (MCWDT) 0 | srss_0_mcwdt_0 |
| □ Multi-Counter Watchdog Timer (MCWDT) 1 | srss_0_mcwdt_1 |
| □ Multi-Counter Watchdog Timer (MCWDT) 2 | srss_0_mcwdt_2 |
| □ Real Time Clock (RTC) | srss_0_rtc_0 |

| Name | Value |
|---|---|
| Overview | |
| Configuration Help | Open EVTGEN Documentation |
| Counter | |
| Reference Clock | CLK_HF3 (18 MHz ± 1%) |
| Low Frequency Clock | CLK_LF (32.768 kHz ± 0.015%) |
| Counter Tick | 1000000 |
| Ratio Control Mode | Hardware Control |
| Ratio Value | 30.51757813 |
| Ratio Dynamic Mode | RatioDynamicMode_0 |
| Comparator0 | |
| Comparator1 | |
| Comparator2 | |
| Comparator3 | |
| Comparator4 | |
| Comparator5 | |
| Comparator6 | |
| Comparator7 | |
| Comparator8 | |
| Comparator9 | |
| Comparator10 | |
| Comparator11 | |
| Comparator12 | |
| Enable | ☑ |
| Trigger Output | ● 12-bit SAR ADC 0 tr_sar_gen_in[0] (ADC) [USED] |
| Work Mode | Active |
| Trigger Mode | Edge Sensitive |
| Active Comparator Value | 1000000 |
| Period Active Event | 1.00000000 |
| Comparator13 | |
| Comparator14 | |
| Comparator15 | |

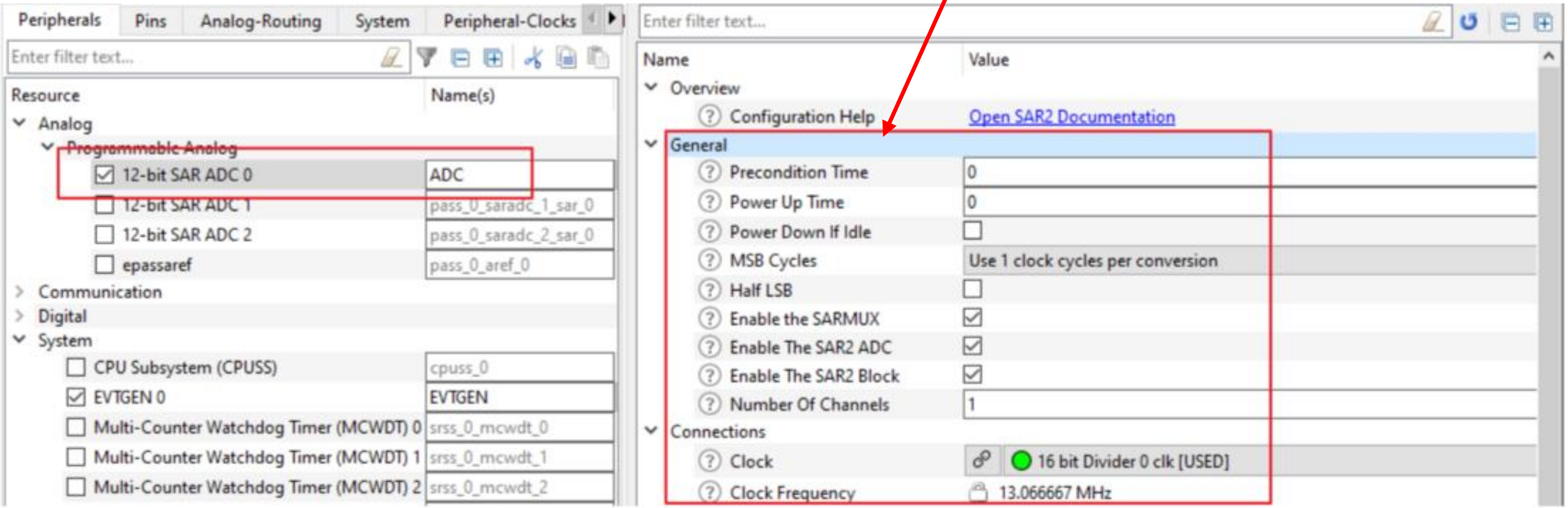
Event Generator setting in this example

You can change the conversion cycle
(Default one second)

# Implementation (contd.)

## ADC general configuration

ADC general and clock setting in this example

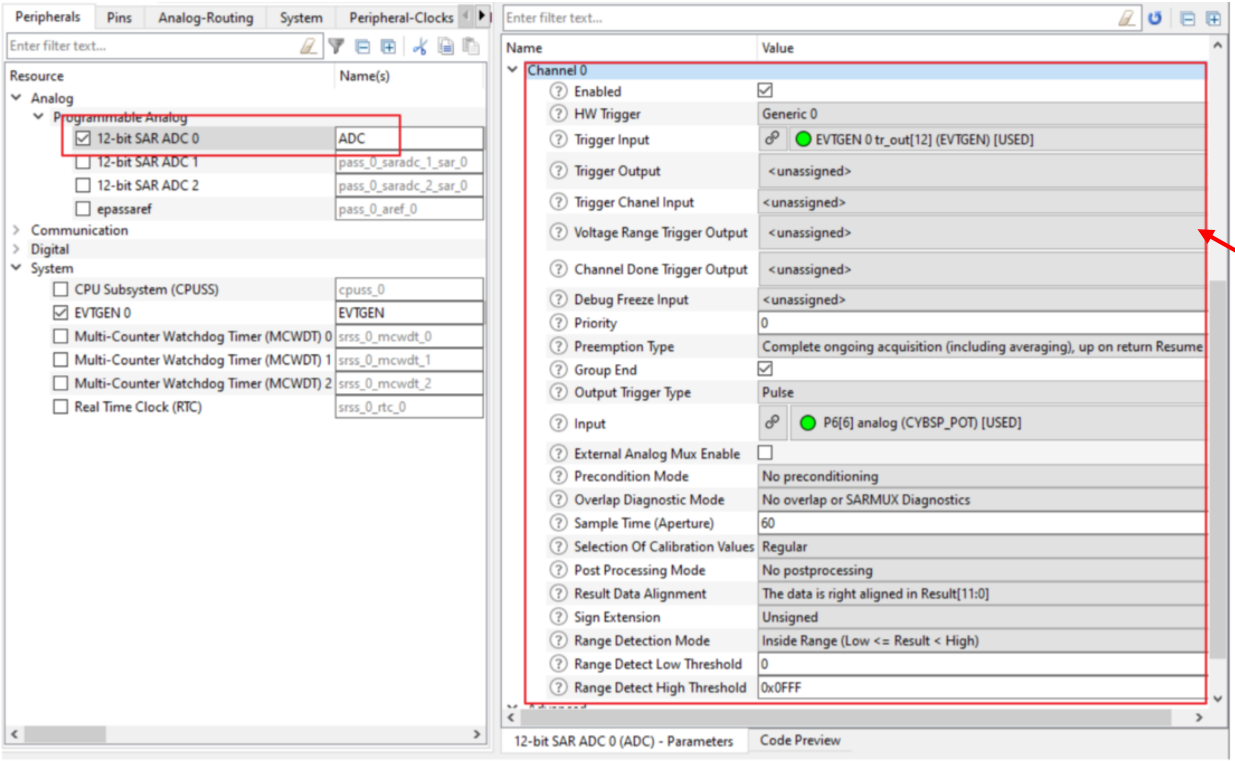


| Resource | Name(s) |
|---|---|
| Analog | |
| Programmable Analog | |
| ☑ 12-bit SAR ADC 0 | ADC |
| □ 12-bit SAR ADC 1 | pass_0_saradc_1_sar_0 |
| □ 12-bit SAR ADC 2 | pass_0_saradc_2_sar_0 |
| □ epassaref | pass_0_aref_0 |
| Communication | |
| Digital | |
| System | |
| □ CPU Subsystem (CPUSS) | cpuss_0 |
| ☑ EVTGEN 0 | EVTGEN |
| □ Multi-Counter Watchdog Timer (MCWDT) 0 | srss_0_mcwdt_0 |
| □ Multi-Counter Watchdog Timer (MCWDT) 1 | srss_0_mcwdt_1 |
| □ Multi-Counter Watchdog Timer (MCWDT) 2 | srss_0_mcwdt_2 |

| Name | Value |
|---|---|
| Overview | |
| Configuration Help | Open SAR2 Documentation |
| General | |
| Precondition Time | 0 |
| Power Up Time | 0 |
| Power Down If Idle | □ |
| MSB Cycles | Use 1 clock cycles per conversion |
| Half LSB | □ |
| Enable the SARMUX | ☑ |
| Enable The SAR2 ADC | ☑ |
| Enable The SAR2 Block | ☑ |
| Number Of Channels | 1 |
| Connections | |
| Clock | ● 16 bit Divider 0 clk [USED] |
| Clock Frequency | 13.066667 MHz |

# Implementation (contd.)

## ADC channel configuration



| Resource | Name(s) |
|---|---|
| Analog | |
| Programmable Analog | |
| ☑ 12-bit SAR ADC 0 | ADC |
| □ 12-bit SAR ADC 1 | pass_0_saradc_1_sar_0 |
| □ 12-bit SAR ADC 2 | pass_0_saradc_2_sar_0 |
| □ epassaref | pass_0_aref_0 |
| Communication | |
| Digital | |
| System | |
| □ CPU Subsystem (CPUSS) | cpuss_0 |
| ☑ EVTGEN 0 | EVTGEN |
| □ Multi-Counter Watchdog Timer (MCWDT) 0 | srss_0_mcwdt_0 |
| □ Multi-Counter Watchdog Timer (MCWDT) 1 | srss_0_mcwdt_1 |
| □ Multi-Counter Watchdog Timer (MCWDT) 2 | srss_0_mcwdt_2 |
| □ Real Time Clock (RTC) | srss_0_rtc_0 |

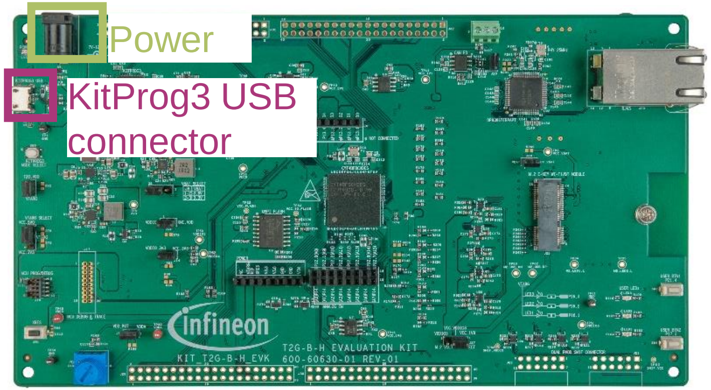
| Name | Value |
|---|---|
| Channel 0 | |
| Enabled | ☑ |
| HW Trigger | Generic 0 |
| Trigger Input | ● EVTGEN 0 tr_out[12] (EVTGEN) [USED] |
| Trigger Output | <unassigned> |
| Trigger Chanel Input | <unassigned> |
| Voltage Range Trigger Output | <unassigned> |
| Channel Done Trigger Output | <unassigned> |
| Debug Freeze Input | <unassigned> |
| Priority | 0 |
| Preemption Type | Complete ongoing acquisition (including averaging), up on return Resume |
| Group End | ☑ |
| Output Trigger Type | Pulse |
| Input | ● P6[6] analog (CYBSP_POT) [USED] |
| External Analog Mux Enable | □ |
| Precondition Mode | No preconditioning |
| Overlap Diagnostic Mode | No overlap or SARMUX Diagnostics |
| Sample Time (Aperture) | 60 |
| Selection Of Calibration Values | Regular |
| Post Processing Mode | No postprocessing |
| Result Data Alignment | The data is right aligned in Result[11:0] |
| Sign Extension | Unsigned |
| Range Detection Mode | Inside Range (Low <= Result < High) |
| Range Detect Low Threshold | 0 |
| Range Detect High Threshold | 0x0FFF |

ADC trigger setting in this example:

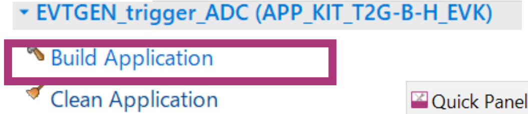
- Trigger input: EVTGEN 0
- Analog Input: P6_6

# Compiling and programming

1. Connect to power and USB cable.
2. Use Eclipse IDE for ModusToolbox™ software for compiling and programming.



3. Compile
   a) Select the target application project in the Project Explorer.
   b) In the Quick Panel, scroll down and click “Build Application” in EVTGEN_trigger_ADC (APP_KIT-T2G-B-H-EVK).

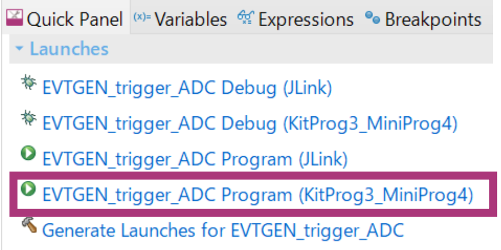


EVTGEN_trigger_ADC (APP_KIT_T2G-B-H_EVK)

Build Application

Clean Application

4. Open a terminal program and select the KitProg3 COM port. Set the serial port parameters to 8N1 and 115200 baud.
5. Programming
   a) Select the target application project in the Project Explorer.
   b) In the Quick Panel, scroll down and click “EVTGEN_trigger_ADC Program (KitProg3_MiniProg4)” in Launches.



Quick Panel | Variables | Expressions | Breakpoints

Launches

EVTGEN_trigger_ADC Debug (JLink)

EVTGEN_trigger_ADC Debug (KitProg3_MiniProg4)

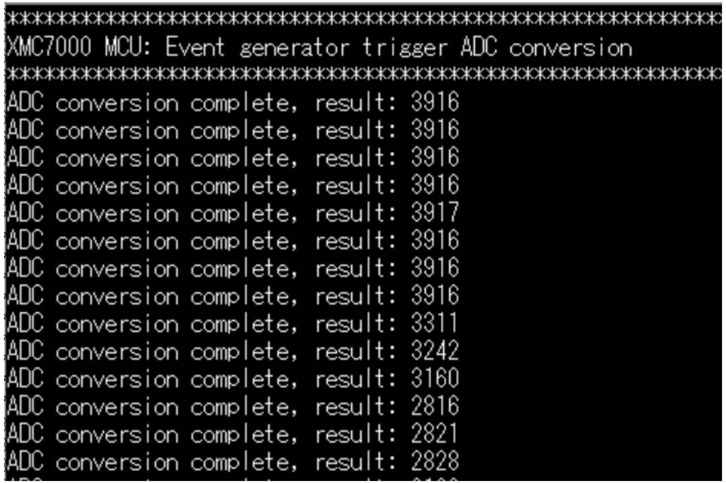
EVTGEN_trigger_ADC Program (JLink)

EVTGEN_trigger_ADC Program (KitProg3_MiniProg4)

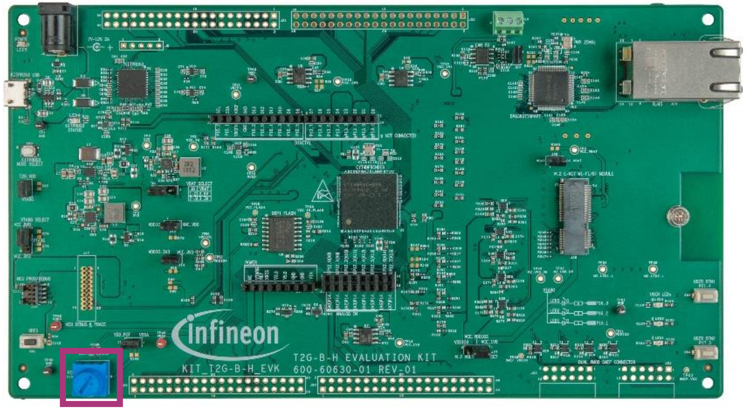
Generate Launches for EVTGEN_trigger_ADC

# Run and test

1. After programming, the application starts automatically. Ensure that input voltages are provided at the analog input pin P6.6. This pin is connected to the potentiometer.
2. Rotate the potentiometer to change the ADC input voltage, and the result prints out every second in the terminal window.
3. Confirm that the input voltages are displayed on the UART terminal.



```
*****************************************************************
XMC7000 MCU: Event generator trigger ADC conversion
*****************************************************************
ADC conversion complete, result: 3916
ADC conversion complete, result: 3916
ADC conversion complete, result: 3916
ADC conversion complete, result: 3916
ADC conversion complete, result: 3917
ADC conversion complete, result: 3916
ADC conversion complete, result: 3916
ADC conversion complete, result: 3916
ADC conversion complete, result: 3311
ADC conversion complete, result: 3242
ADC conversion complete, result: 3160
ADC conversion complete, result: 2816
ADC conversion complete, result: 2821
ADC conversion complete, result: 2828
```



Potentiometer

# References

**Datasheet**

- CYT4BF datasheet 32-bit Arm® Cortex®-M7 microcontroller TRAVEO™ T2G family

**Architecture technical reference manual**

- TRAVEO™ T2G automotive body controller high family architecture technical reference manual

**Registers technical reference manual**

- TRAVEO™ T2G automotive body controller high registers technical reference manual

**PDL/HAL**

- PDL
- HAL

**Training**

- TRAVEO™ T2G Training

# Revision History

| Revision | ECN | Submission Date | Description of Change |
|---|---|---|---|
| ** | 7840285 | 2022/11/24 | Initial release. |

# Important notice and warnings

All referenced product or service names and trademarks are the property of their respective owners.

**Edition 2022-11**
**Published by**
**Infineon Technologies AG**
**81726 Munich, Germany**

**Do you have a question about this document?**
**Go to:**
**www.infineon.com/support**

**Document reference**
**002-36103 Rev. ****

**IMPORTANT NOTICE**

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics (“Beschaffenheitsgarantie”) .

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer’s compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer’s products and any use of the product of Infineon Technologies in customer’s applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer’s technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (**www.infineon.com**).

**WARNINGS**

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies’ products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.