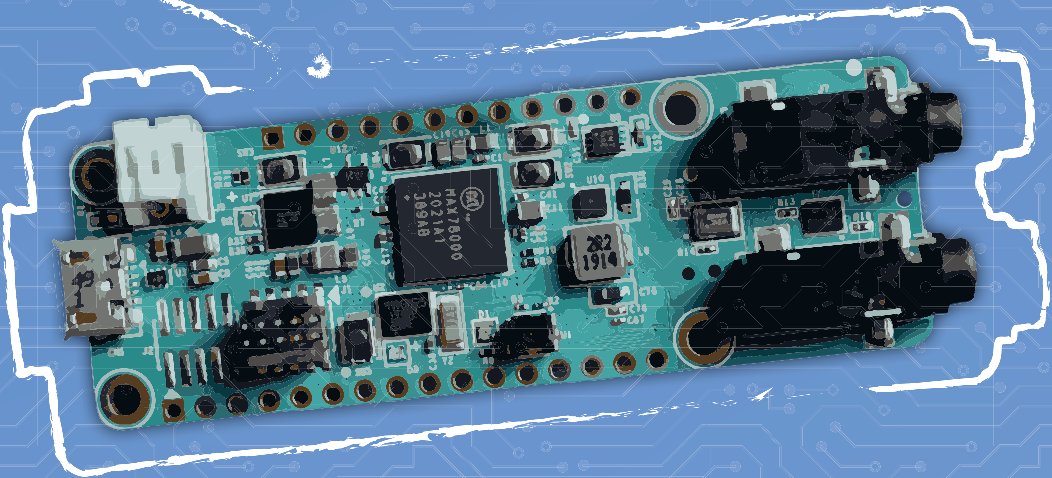
H0W2 Volume 2

Get Started with the MAX78000FTHR Development Board

Build your own AI microcontroller applications from scratch



Dogan Ibrahim

maxim integrated

elektor knows how

How2: Get Started with the MAX78000FTHR Development Board

Dogan Ibrahim

elektor
design > share > sell

an Elektor Publication

● This is an Elektor Publication. Elektor is the media brand of
Elektor International Media B.V.
78 York Street
London W1H 1DP, UK
Phone: (+44) (0)20 7692 8344
© Elektor International Media BV 2021
First published in the United Kingdom 2021

● British Library Cataloguing in Publication Data
Catalogue record for this book is available from the British Library

● ISBN: 978-3-89576-447-9

● EISBN: 978-3-89576-448-6

Prepress production: DMC ¦ dave@daverid.com
Printed in the Netherlands by Ipskamp

To my wife Nadire, my daughter Alev, and my son Ahmet, for their love and wisdom.

• Declaration

The author and publisher have used their best efforts in ensuring the correctness of the information contained in this book. They do not assume, or hereby disclaim, any liability to any party for any loss or damage caused by errors or omissions in this book, whether such errors or omissions result from negligence, accident, or any other cause.

The author expresses his sincere thanks to **Maxim Integrated** for permitting to include various tables, figures, and program codes in this book.

Additionally, the author would like to thank Mr. Ole Dreessen (Principal MTS, Field Applications of Maxim Integrated, Munich) and his team for reviewing the manuscript.

● Preface

A microcontroller is a single-chip microprocessor system that contains data and program memory, serial and parallel I/O, timers, external and internal interrupts, all integrated into a single chip that can be purchased for as little as $2.00. About 40% of microcontroller applications are in office automation, such as PCs, laser printers, fax machines, intelligent telephones, and so forth. About one-third of microcontrollers are found in consumer electronic goods. Products like CD and DVD players, hi-fi equipment, video games, washing machines, cookers, and so on fall into this category. The communications, automotive, and military markets share the rest of the application areas.

The MAX78000FTHR (from Maxim Integrated) is a small development board based on a MAX78000 microcontroller unit (MCU). This MCU is targeted at artificial intelligence (AI) applications running at the edge of the technology. AI applications require large amounts of processing power and memory. This is why the MAX78000 combines an Arm Cortex-M4 processor with a floating-point unit (FPU), convolutional neural network (CNN) accelerator, and a RISC-V core into a single device. The MAX78000 microcontroller also has the important feature that it is designed for ultra-low power consumption, thus making it ideal in many AI-based portable applications.

The MAX78000 microcontroller uses a Convolutional Neural Network (CNN) based approach. A CNN is a special kind of neural network where convolution is used at its heart. Convolution is a kind of matched filtering and is often used in signal and image processing. Since CNNs are particularly used in these applications, the MAX78000FTHR development board integrates an onboard camera and microphone interfaced with I2S for streaming audio applications.

The board is recognised as a USB device when connected to a computer. It comes preloaded with an audio keyword spotting (KWS) demo, where the flashing rate of an onboard LED can be changed by speech commands. When it detects the word "Go" the demo enters number recognition mode in which it will blink an LED the number of times as commanded by the speaker. In other words, after saying "Four", the LED will blink four times. "Stop" returns to normal mode.

The MAX78000FTHR development board includes the following peripheral devices on-board:

- Tiny VGA camera
- Digital microphone
- microSD card slot
- 1 MB RAM
- SWD debugger/programmer over USB
- LiPo battery charger
- RGB LEDs and user pushbuttons for projects

An Eclipse-based SDK is provided by Maxim that can be used to develop programs for the MAX78000FTHR. The development environment contains the Eclipse IDE, MinGW, GCC toolchains for Arm and RISC-V processors, OpenOCD, and a few other utilities. Additionally, libraries and many example programs are provided to help the starters develop projects.

This book is project-based and its main aim has been to teach the basic features of the MAX78000FTHR development board and show how it can be used in various projects. Many fully tested projects are given in the book, where each project is described fully and in detail, and the complete program listings are given for each project. Readers should be able to use the projects as they are, or modify them to suit their own needs. The following sub-headings are used while describing each project:

- Description of the project
- Aim of the project
- Background (if applicable)
- Block diagram
- Circuit diagram
- Program listing
- Suggestions for future work (if applicable)

Knowledge of C will be useful to readers. Also, familiarity with at least one microcontroller development board (preferably with an Arm processor) will be an advantage. Knowledge of assembly language programming is not required because all projects in the book are based on using C. Some knowledge of the theory of neural networks will make it easier for the readers to follow the chapter on Convolutional Neural Networks.

This book is written for students, practising engineers, and for hobbyists interested in developing artificial interface applications using the MAX78000FTHR development board. An attempt has been made to include as many projects as possible, limited only by the size of the book.

Dogan Ibrahim
London, 2021

● Table of Contents

Chapter 1 ● The MAX78000FTHR Development Board

1.1 ● Overview

The MAX78000FTHR is an advanced ultra-low-power microcontroller development board, created to help implement artificial intelligence (AI) solutions. Based on the Arm Cortex-M4F processor, the MAX78000 includes an integrated Convolution Neural Network (CNN) accelerator. In this chapter, we will be looking at the basic hardware details of this board.

1.2 ● Basic features

The MAX78000FTHR development board (see Figure 1.1 for top view, and Figure 1.2 for bottom view) has the following basic features:

- Arm Cortex-M4 100MHz processor with FPU and RISC-V core
- 32-bit RISC-V coprocessor
- 512KB Flash memory
- 128KB SRAM
- 16KB Cache
- Convolution Neural Network Accelerator (CNN)
- 12-bit camera interface (parallel)
- Wearable PMIC (MAX20303)
- On-board DAPLink Debug and programming
- Micro-USB connector
- Micro SD card holder
- SPI, I2C, LPUART, I2S interfaces
- LPTIMER and analog comparators
- Breadboard compatible pins (dual-row header)
- Digital microphone (Knowledge Acoustics SPH0645LM4H-B)
- 3 x RGB indicator LEDs
- 5 x pushbuttons
- CMOS VGA image sensor (Omnivision OVM7692-RYAA)
- Stereo audio CODEC (Maxim MAX9867)
- Virtual UART console
- 10-pin Cortex Debug Header for RISC-V Coprocessor
- Li-Ion battery charger (battery not included)

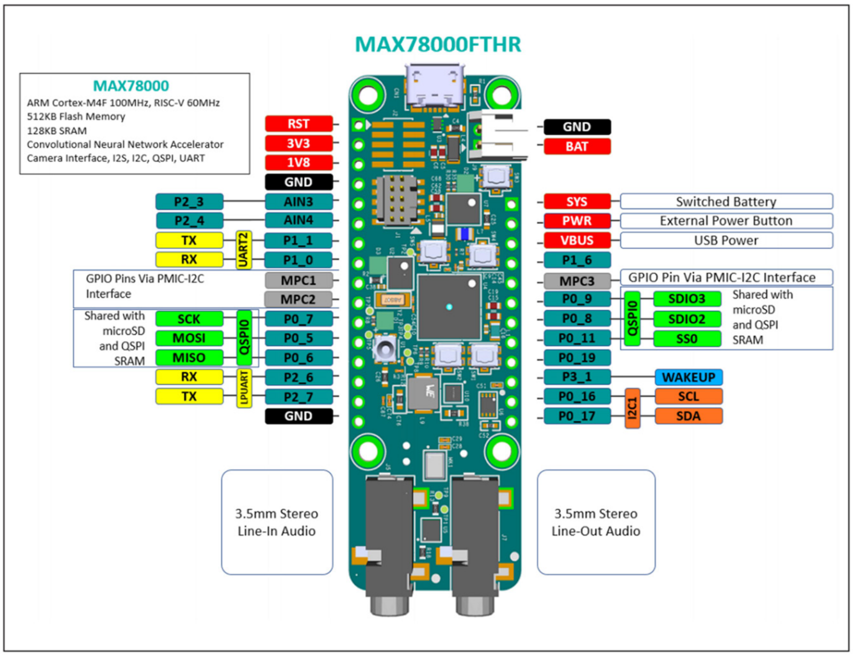
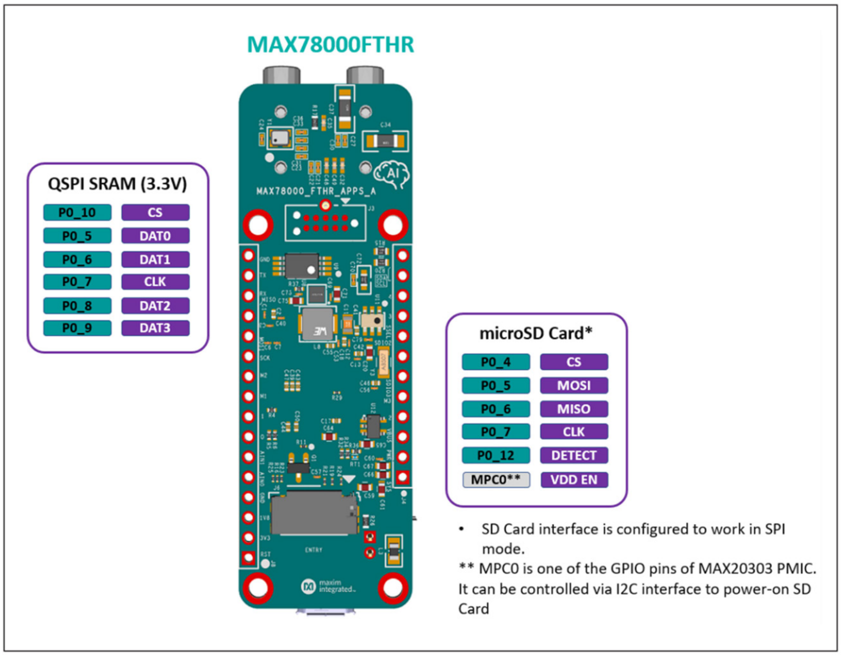

MAX78000FTHR
MAX78000
ARM Cortex-M4F 100MHz, RISC-V 60MHz
512KB Flash Memory
128KB SRAM
Convolutional Neural Network Accelerator
Camera Interface, I2S, I2C, QSPI, UART
RST
3V3
1V8
GND
P2_3 AIN3
P2_4 AIN4
TX P1_1
RX P1_0
UART2
GPIO Pins Via PMIC-I2C Interface
MPC1
MPC2
Shared with microSD and QSPI SRAM
SCK P0_7
MOSI P0_5
MISO P0_6
QSPI0
RX P2_6
TX P2_7
LPUART
GND
GND
BAT
SYS Switched Battery
PWR External Power Button
VBUS USB Power
P1_6
MPC3 GPIO Pin Via PMIC-I2C Interface
P0_9 SDIO3
P0_8 SDIO2
P0_11 SS0
QSPI0
Shared with microSD and QSPI SRAM
P0_19
P3_1 WAKEUP
P0_16 SCL
P0_17 SDA
I2C1
3.5mm Stereo Line-In Audio
3.5mm Stereo Line-Out Audio

Figure 1.1 Top view of the MAX78000FTHR board



- SD Card interface is configured to work in SPI mode.

** MPC0 is one of the GPIO pins of MAX20303 PMIC. It can be controlled via I2C interface to power-on SD Card

Figure 1.2 Bottom view of the MAX78000FTHR board

1.3 ● Pushbuttons and LEDs

There are 5 pushbuttons (named **SW1** to **SW5**) and 3 RGB LEDs (named **D1** to **D3**) on the board as shown in Figure 1.3.

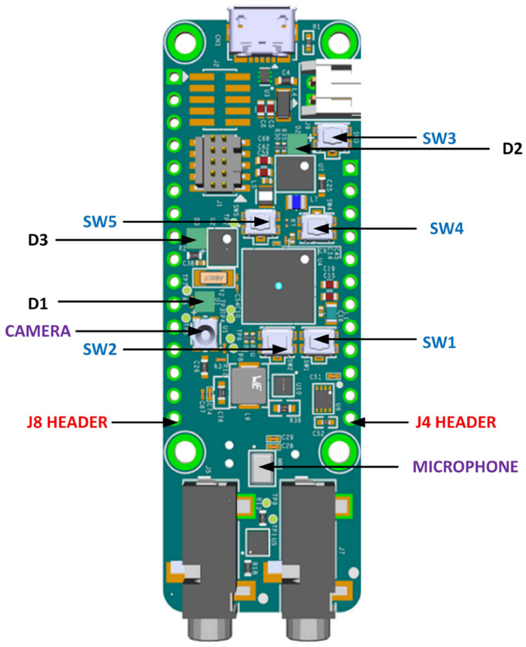


Figure 1.3 Pushbuttons and LEDs on the board

The functions of the pushbuttons are:

SW1: User-programmable function button connected to the MAX78000 Port 0_2 through a debouncer IC.

SW2: User-programmable function button connected to the MAX78000 Port 1_7 through a debouncer IC.

SW3: PMIC Power Button. When the board is in a powered-on state, pressing this button for 12 seconds performs a hard power-down. When the board is in a powered-off state, pressing this button powers on the board. This button can also be read by the MAX78000 firmware, PMIC_PFN2 signal connected to Port 3_1 is a buffered input of the button status. When the button is pressed, this signal goes to a logic-low state.

SW4: Resets the MAX78000 through the RSTN input of the MAX78000.

SW5: DAPLink adapter button. Keep this button pressed while applying power to the board to put the MAX32625 DAPLink adapter onboard to MAINTENANCE mode for DAPLink firmware updates.

The functions of the LEDs are:

D1: Connected to the MAX78000 GPIO ports. This LED can be controlled by user firmware.

> **Port 2_0 : Red**
> **Port 2_1 : Green**
> **Port 2_2 : Blue**

D2: Connected to MAX20303 PMIC LEDx outputs. These LEDs can be controlled through I2C commands. They also can be configured as charge status indicators by issuing I2C commands.

D3: DAPLink adapter MAX32625 status LED. Controlled by the DAPLink adapter and cannot be used as a user LED.

1.4 • GPIO Pinout

There are two headers at either side of the board where GPIO signals are terminated. As shown in Figure 1.3, **Header J4** has 12 pins and **Header J8** has 16 pins. The board includes the following GPIO port pins:

> **PORT0**: P0_5, P0_6, P0_7, P0_8, P0_9, P0_11, P0_16, P0_17, P0_19
> **PORT1**: P1_0, P1_1
> **PORT2**: P2_3, P2_4, P2_6, P2_7
> **PORT3**: P3_1

Tables 1.1 and 1.2 show the J4 and J8 pin names and their descriptions. Notice some pins are shared. For example, P1_0 and p1_1 are shared with the UART RX and TX pins respectively. Similarly, P2_3 and P2_4 are shared with analog inputs AIN3 and AIN4 respectively.

PIN	NAME	DESCRIPTION
1	SYS	SYS Switched Connection to the Battery. This is the primary system power supply and automatically switches between the battery voltage and the USB supply when available.
2	PWR	Turns off the PMIC if shorted to Ground for 13 seconds. Hard power-down button.
3	VBUS	USB VBUS Signal. This can be used as a 5V supply when connected to USB. This pin can also be used as an input to power the board, but this should only be done when not using the USB connector since there is no circuitry to prevent current from flowing back into the USB connector.
4	P1_6	GPIO
5	MPC3	GPIO controlled by PMIC through the I^{2}C interface. Open drain or push-pull programmable.
6	P0_9	GPIO or QSPI0 SDIO3 signal. Shared with SD card and on-board QSPI SRAM.
7	P0_8	GPIO or QSPI0 SDIO2 signal. Shared with SD Card and on-board QSPI SRAM.
8	P0_11	GPIO or QSPI0 slave select signal.
9	P0_19	GPIO
10	P3_1	GPIO or Wake-up signal. This pin is 3.3V only.
11	P0_16	GPIO or I2C1 SCL signal. An on-board level shifter allows selecting 1.8V or 3.3V operation through R15 or R20 resistors. Do not populate both.
12	P0_17	GPIO or I2C1 SDA signal. An on-board level shifter allows selecting 1.8V or 3.3V operation through R15 or R20 resistors. Do not populate both.

Table 1.1 Header J4 pins

PIN	NAME	DESCRIPTION
1	RST	Master Reset Signal
2	3V3	3.3V Output. Typically used to provide 3.3V to peripherals connected to the expansion headers
3	1V8	1.8V Output. Typically used to provide 1.8V to peripherals connected to the expansion headers
4	GND	Ground
5	P2_3	GPIO or Analog Input (AIN3 channel)
6	P2_4	GPIO or Analog Input (AIN4 channel)
7	P1_1	GPIO or UART2 Tx signal
8	P1_0	GPIO or UART2 Rx signal
9	MPC1	GPIO controlled by PMIC through I^{2}C interface. Open drain or push-pull programmable
10	MPC2	GPIO controlled by PMIC through I^{2}C interface. Open drain or push-pull programmable
11	P0_7	GPIO or QSPI0 clock signal. Shared with SD card and on-board QSPI SRAM
12	P0_5	GPIO or QSPI0 MOSI signal. Shared with SD card and on-board QSPI SRAM
13	P0_6	GPIO or QSPI0 MISO signal. Shared with SD card and on-board QSPI SRAM
14	P2_6	GPIO or LPUART Rx signal
15	P2_7	GPIO or LPUART Tx signal
16	GND	Ground

Table 1.2 Header J8 pins

1.5 ● The FTHR board component interface signals

Figure 1.4 shows the MAX78000FTHR Application Platform. The figure shows the application interface signals, buttons, and LEDs for:

- RISC-V JTAG
- Camera module
- PMIC
- Digital microphone
- Audio CODEC
- UART
- RGB LED
- Buttons

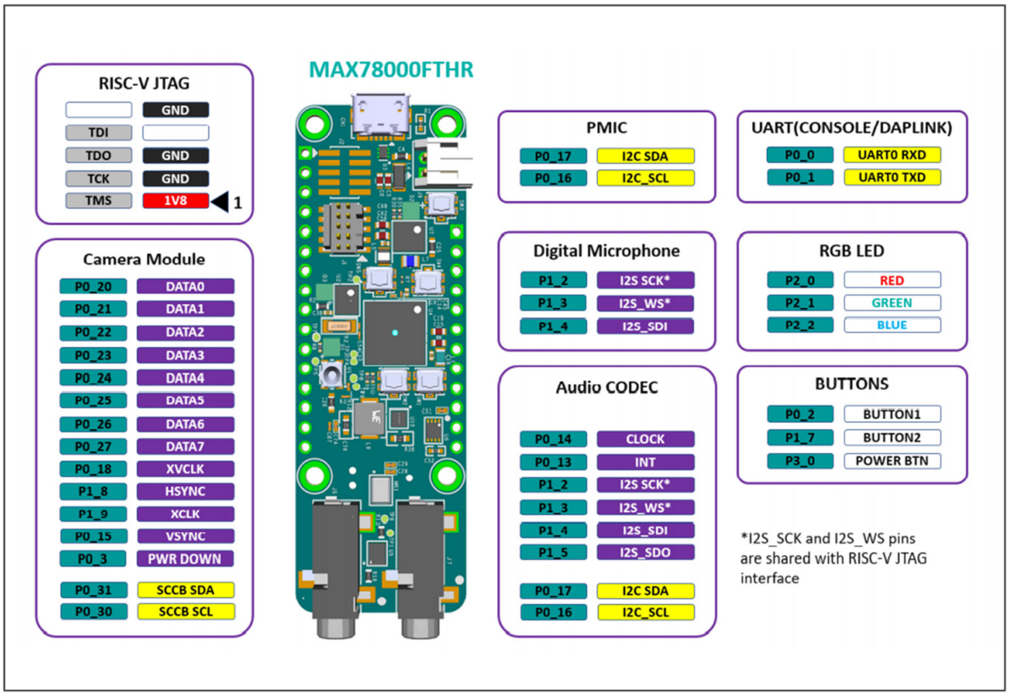


Figure 1.4 The application platform

1.6 ● The startup and the demo application

The MAX78000FTHR is pre-programmed with the **Audio Keyword Spotting** demo application. This demo application is useful for checking if the board is working correctly.

The steps to startup the MAX78000FTHR and the demo application are:

- Connect the MAX78000FTHR to the USB port of your PC using a micro-USB cable
- RGB LED (**D2**) will turn ON green to indicate that the pre-programmed demo application **Audio Keyword Spotting** is running
- The on-board microphone (see Figure 1.3) starts listening for the keyword GO
- When the keyword **GO** is detected, RGB LED (**D2**) turns yellow
- In this mode, when one of nine keywords is detected, the RGB LED (**D1**) blinks blue one to nine times based on the number detected by the convolutional neural network. For example, speak the word **FOUR** and the blue LED blinks 4 times
- The STOP command exits number keyword detection, and the RGB LED (**D2**) turns on green again, and RGB LED (**D1**) turns off

1.7 ● The voltage regulator/battery charger

The MAX20303 PMIC chip is used to power the MAX78000FTHR board and also to charge a Li-Ion battery (not included). The MAX20303 has an internal MOSFET that connects the battery to system output. The smart power selector unit inside the PMIC seamlessly controls power to the board when a battery is present and when the board is connected to a USB power source (for more information, see document: *MAX78000FTHR Application Platform, Rev. 11/20, Maxim Integrated Products Inc.* at link: https://datasheets.maximintegrated.com/en/ds/MAX78000FTHR.pdf).

1.8 ● DAP-link/SWD debug interface

The onboard MAX32625 microcontroller is pre-programmed with DAPLink firmware. It allows the debugging and programming of the MAX78000 Arm core through a USB interface. A standard 10-pin DAP-link/SWD interface (header J2) is provided as shown in Figure 1.5 with the pin configuration shown in Figure 1.6. DAPLink adapter button SW5 must be kept pressed while applying power to the board to put the MAX32625 DAPLink adapter onboard to MAINTENANCE mode for DAPLink firmware updates (see link: https://os.mbed.com/teams/MaximIntegrated/wiki/MAX32625PICO-Firmware-Updates). The DAPLink adapter status LED D3 is controlled by the DAPLink adapter. The board comes pre-installed with a bootloader enabling driverless drag-n-drop updates. This bootloader can be used to update or restore the DAPLink firmware on the MAX32625, or to load your custom application on the board. To activate the bootloader, simply hold the SW5 button down while applying power to the board. When the bootloader detects the button press at power on, it will connect to the PC as a drive named "MAINTENANCE". Simply Drag-n-Drop the desired image onto the MAINTENANCE drive to load new firmware into the board.

We can use a virtual com port and send data from the development board to a terminal emulation program (e.g. Putty) using e.g. printf statements. This can be very helpful during debugging a program.

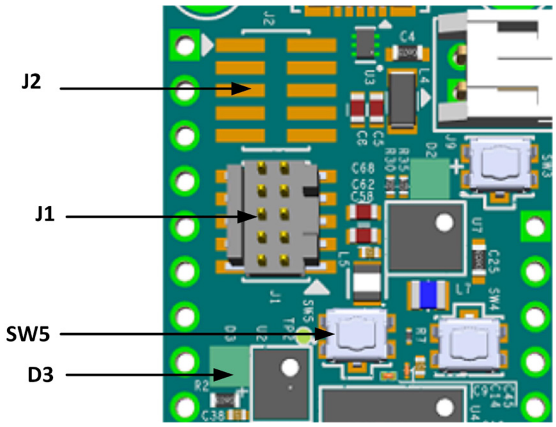


Figure 1.5 Headers J1 and J2

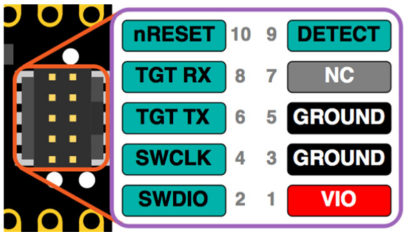


Figure 1.6 Header J2 pin configuration

A standard 10-pin JTAG header J1 (Figure 1.5) allows debugging and programming the RISC-V core of the MAX78000.

Chapter 2 • The MAX78000 Microcontroller

2.1 ● Overview

The MAX78000 is an advanced ultra-low-power microcontroller, developed to help implement artificial intelligence (AI), as well as standard embedded microcontroller solutions. Among the Arm Cortex-M4F processor, the MAX78000 includes an integrated Convolutional Neural Network (CNN) accelerator. In this chapter, we will be looking at some of the basic hardware details of this microcontroller (much of this Chapter has been taken from the **MAX78000 User Guide: MAX78000 User Guide, UGXXXX; Rev 0.5; 08/31/2020**)

2.2 ● Basic features of the MAX78000

One of the problems with developing Artificial Intelligence (AI) based products is the requirement for extreme computational power. This can only be achieved by having a fast processor in addition to using a processor dedicated to AI. The MAX78000 is a new AI microcontroller built to enable neural network applications to be executed with ultra-low-power and ease of use.

The MAX78000 is an advanced microcontroller featuring an Arm Cortex-M4 with FPU CPU for efficient system control and a Convolutional Neural Network (CNN) accelerator. The CNN has a weight storage SRAM memory of 442 KB, and can support 1-, 2-, 4-, and 8-bit weights, enabling AI network updates to be made on the fly. Additionally, the CNN engine has 512 KB of data memory, and because of its highly flexible architecture, it allows networks to be trained in conventional toolsets (e.g. PyTorch and TensorFlow) and converted for execution on the MAX78000 using Maxim tools.

In addition to the memory in the CNN engine, the MAX78000 has large 512 KB flash core memory and 128 KB SRAM. The microcontroller supports high-speed communication interfaces, such as I2S, I2C, SPI, UART, camera interface, and so on.

The MAX78000 microcontroller can be used in the following applications where high processing speeds are required:

- Audio processing (e.g. sound classification, noise cancellation, multi-keyword recognition, etc)
- Facial recognition
- Time-series data processing (heart rate signal analysis, health signal analysis, multi-sensor analysis, etc)

The MAX78000 microcontroller is available in 81-pin and 130-pin packages. Figure 2.1 shows the simplified internal structure of the MAX78000 microcontroller. On the top right of the figure is the CNN engine communicating over the internal multi-layer bus matrix. RISC-V and the clock circuitry are located on the top left of the figure. Below the CNN engine we can see the peripheral control devices which consist of the following modules:

- I2S master/slave
- 3 x I2C
- 3 x UART and 1 x LPUART
- 2 x SPI
- 1-wire master
- 4 x pulse train engines
- 4 x 32-bit timers and wake-up timer
- 2 x 32-bit LPTIMERS
- 1 x parallel camera interface

In the middle left part of the figure we can see the various memory modules, including the cache and boot memories. The bottom left part of the figure is where voltage regulation, dynamic voltage scaling, and power control are handled. Finally, at the bottom part of the figure, we can see the DMA, security modules, ADC, and comparator modules.

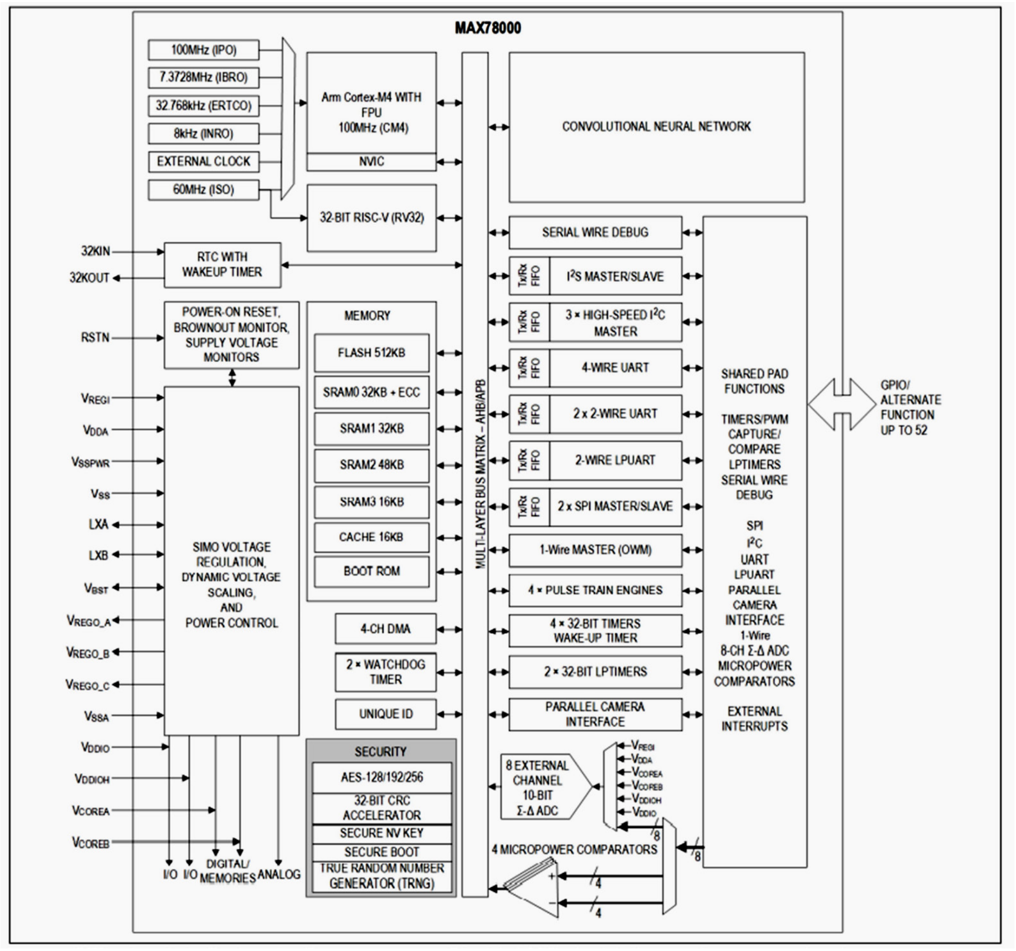


Figure 2.1 Internal structure of the MAX78000 microcontroller

2.2.1 ● The Convolutional Neural Network Accelerator (CNN)

The CNN engine is at the heart of the MAX78000 and it consists of 64 parallel processors with 512 KB of SRAM. Each of the processors includes a pooling unit and a convolutional engine with dedicated weight memory, where 4 processors share one 32KB data memory. These are further organised into groups of 16 processors that share common controls. A group of 16 processors operates as a slave to another group or independently. Data is read from SRAM associated with each processor and written to any data memory located within the accelerator. Any given processor has the visibility of its dedicated weight memory and to the data memory instance it shares with the three others.

Further information on the CNN engine features can be obtained from the document: **MAX78000 Ultra-Low-Power Arm Cortex-M4 Processor with FPU-Based Microcontroller with Convolutional Neural Network Accelerator**, located on the website:

https://www.maximintegrated.com/en/products/microcontrollers/MAX78000.html

2.2.2 ● The memory

The internal flash memory providing non-volatile storage of programs and data is 512 KB. Internal SRAM is 128 KB that provides the retention of application data in all power modes except POWER DOWN. The SRAM is divided into 4 banks (see Figure 2.1): SRAM0 and SRAM1 are both 32 KB, SRAM2 is 48 KB and SRAM3 is 16 KB.

2.2.3 ● Comparators

The eight AIN[7:0] inputs can be configured as four pairs and deployed as four independent comparators with the following features:

- Comparison events can trigger interrupts
- Events can wake the CM4 from SLEEP, LOW POWER, MICRO POWER, STANDBY, or BACKUP operating modes
- Can be active in all power modes

2.2.4 ● Clocking

The following can be selected as the clock sources (Figure 2.2):

- Internal primary oscillator (IPO) at a nominal frequency of 100MHz
- Internal secondary oscillator (ISO) at a nominal frequency of 60MHz
- Configurable internal nano-ring oscillator (INRO) at 8kHz, 16kHz, or 30kHz
- External RTC oscillator at 32.768kHz (ERTCO) (external crystal required)
- Internal baud rate oscillator at 7.3728MHz (IBRO)
- External square-wave clock up to 80MHz

There are multiple external clock inputs:

- LPTMR0 and LPTMR1 can be clocked from unique external sources
- I2S can be clocked from its own external source

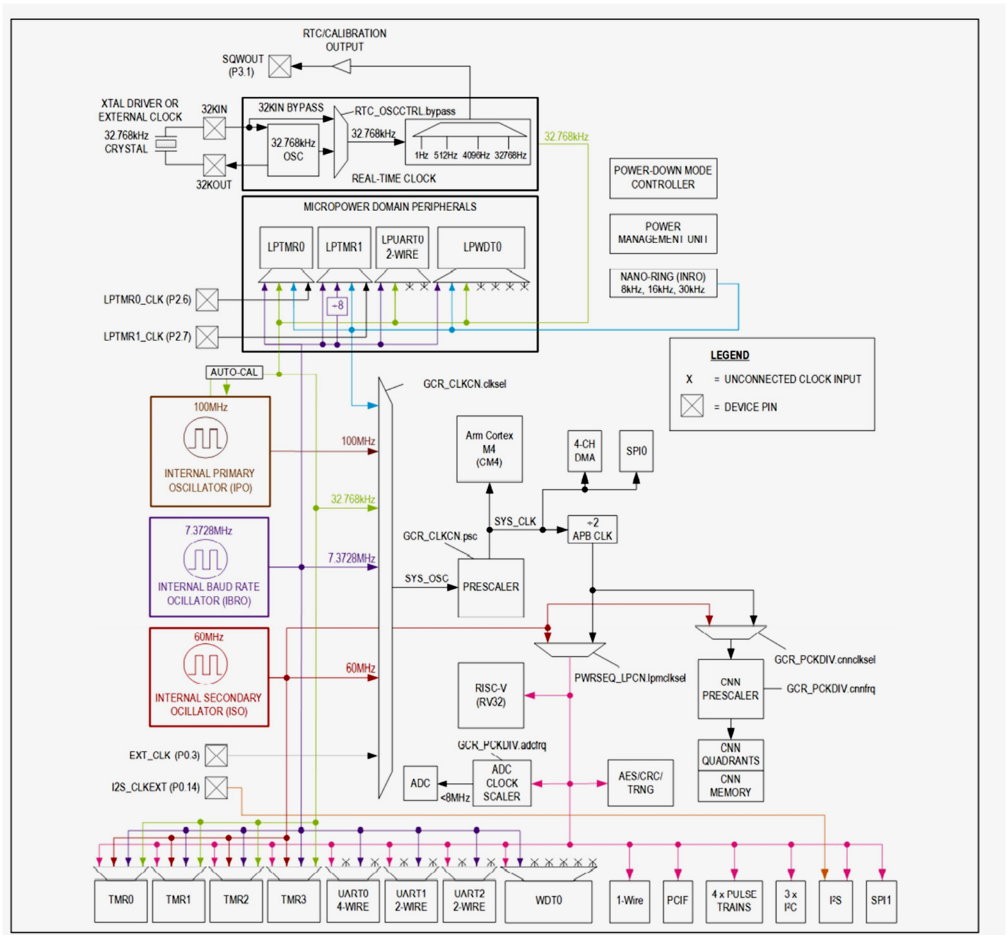


Figure 2.2 Clock sources

2.2.5 ● General-purpose input-output (GPIO) and special function pins

The MAX78000 microcontroller provides up to 52 GPIO pins. Most general-purpose I/O (GPIO) pins share both a firmware-controlled I/O function and one or more alternate functions, associated with peripheral modules. Pins can be individually enabled for GPIO or peripheral special function use.

Configuring a pin as a special function usually supersedes its use as a firmware-controlled I/O. Although this multiplexing between peripheral and GPIO functions is usually static, it can also be done dynamically. In GPIO mode, pins are logically divided into ports of 32 pins. Each pin of a port has an interrupt function that can be independently enabled and configured as a level or edge-sensitive interrupt. All GPIOs of a given port share the same interrupt vector.

When configured as GPIO, the following features are provided, which can be independently enabled or disabled:

- Configurable as input, output, bidirectional, or high impedance
- Optional internal pull-up resistor or internal pull-down resistor when configured as input
- Exit from low-power modes on rising or falling edge
- Selectable standard- or high-drive modes

2.2.6 ● Parallel Camera Interface (PCIF)

The Parallel Camera Interface (PCIF) is a peripheral designed to read data from camera sensors. The PCIF is a low voltage interface suited for CMOS image sensors. It provides up to 12-bits of parallel access capability with single capture and continuous mode operation.

2.2.7 ● Analog-to-Digital Converter (ADC)

The ADC resolution is 10-bits with an 8MHz maximum clock rate, providing an integrated reference generator and single-ended input multiplexer. The multiplexer selects an input channel from one of the eight external analog input signals (AIN0–AIN7) or the internal power supply inputs.

There are two sources for ADC reference voltage:

- Internal 1.22V bandgap
- VSSA analog supply

An optional feature allows samples captured by the ADC to be automatically compared against user-programmable high and low limits. Up to four-channel limit pairs can be configured in this way. The comparison allows the ADC to trigger an interrupt (and potentially wake the CPU from a power mode) when a captured sample goes outside the pre-programmed limit range. Since this comparison is performed directly by the sample limit monitors, it can be performed even while the CPU is in SLEEP, LOW POWER, or MICRO POWER mode. The eight AIN[7:0] inputs of the ADC can be configured as four pairs and deployed as four independent comparators.

In addition to the 8 inputs, the ADC can measure several other voltages (see the MAX78000 datasheet for further information)

2.2.8 ● Power management (PMU)

The power management unit (PMU) provides high-performance operation while minimising power consumption. It exercises intelligent, precise control of power distribution to the CPUs and peripheral circuitry.

The following modes of operation are available (see Table 2.1):

ACTIVE Mode

This is the normal operational mode, all digital and analog peripherals are available on demand. Peripherals not in use can be disabled by dynamic clocking.

SLEEP Mode

This mode consumes less power but wakes faster because the clocks can optionally be enabled.

LOW POWER Mode (LPM)

This mode is suitable for running the RISC-V processor to collect and move data from enabled peripherals.

MICRO POWER Mode (μPM)

This mode is used for extremely low power consumption while using a minimal set of peripherals to provide wake-up capability.

STANDBY Mode

This mode is used to maintain the system operation while keeping time with the RTC.

BACKUP Mode

This mode is used to maintain the system RAM.

POWER DOWN Mode (PDM)

This mode is used during product level distribution and storage. All oscillators are powered down and there is no data retention in this mode, however values in the flash are preserved.

	CM4	RV32	CNN	Periph.	GPIO	DMA	Clocks
Active	ON	ON	ON	ON	ON	ON	All ON
Sleep	Sleep	Sleep	Optional	ON	ON	ON	All ON
Low power	Retained	Retained	ON	ON	ON	Retained	IPO: OFF
Micro power	Retained	Retained	Memory: retained Quadrants: OFF	Retained	Retained	Retained	IPO: OFF ISO: OFF
Standby	Retained	Retained	Memory: retained Quadrants: OFF	Retained	Retained	Retained	IPO: OFF ISO: OFF
Backup	OFF	OFF	Memory: retained Quadrants: OFF	OFF	OFF	OFF	IPO: OFF ISO: OFF IBRO: OFF INRO: OFF
Power down	OFF	OFF	OFF	OFF	OFF	OFF	OFF

Table 2.1 Power management modes comparison

In all operating modes other than ACTIVE, wakeup sources are required to re-enter ACTIVE operation.

2.2.9 ● Real-time clock (RTC)

The Real-Time Clock (RTC) is a 32-bit binary timer that keeps the time of day up to 136 years. It provides time-of-day and sub-second alarm functionality in the form of system interrupts. The 32-bit seconds register can count up to approximately 136 years and be translated to calendar format by application software. The RTC provides a time-of-day alarm that can be programmed to any future value between 1 second and 12 days. When configured for long intervals, the time-of-day alarm can be used as a power-saving timer, allowing the device to remain in an extremely low-power mode, but still awaken periodically to perform assigned tasks. A second independent 32-bit 1/4096 sub-second alarm can be programmed with a tick resolution of 244µs. Both can be configured as recurring alarms. When enabled, either alarm can cause an interrupt or wake the device from most low-power modes. The time base is generated by a 32.768kHz crystal or an external clock source.

2.2.10 ● Programmable timers

32-Bit Timer/Counter/PWM (TMR, LPTMR)

General-purpose, 32-bit timers provide timing, capture/compare, or generation of pulse-width modulated (PWM) signals with minimal software interaction.

The timer provides the following features:

- 32-bit up/down auto-reload
- Programmable prescaler
- PWM output generation
- Capture, compare, and capture/compare capability
- External pin multiplexed with GPIO for timer input, clock gating, or capture