The following document contains information on Cypress products. The document has the series name, product name, and ordering part numbering with the prefix “MB”. However, Cypress will offer these products to new and existing customers with the series name, product name, and ordering part number with the prefix “CY”.

How to Check the Ordering Part Number

1. Go to www.cypress.com/pcn.
2. Enter the keyword (for example, ordering part number) in the **SEARCH PCNS** field and click **Apply**.
3. Click the corresponding title from the search results.
4. Download the Affected Parts List file, which has details of all changes

For More Information

Please contact your local sales office for additional information about Cypress products and solutions.

About Cypress

Cypress is the leader in advanced embedded system solutions for the world's most innovative automotive, industrial, smart home appliances, consumer electronics and medical products. Cypress' microcontrollers, analog ICs, wireless and USB-based connectivity solutions and reliable, high-performance memories help engineers design differentiated products and get them to market first. Cypress is committed to providing customers with the best support and development resources on the planet enabling them to disrupt markets by creating new product categories in record time. To learn more, go to www.cypress.com.

AN205204

FR Family MB91460 Series, Sound Generation

This application note gives an overview off different sound generation possibilities on the MB91460 series MCU's.

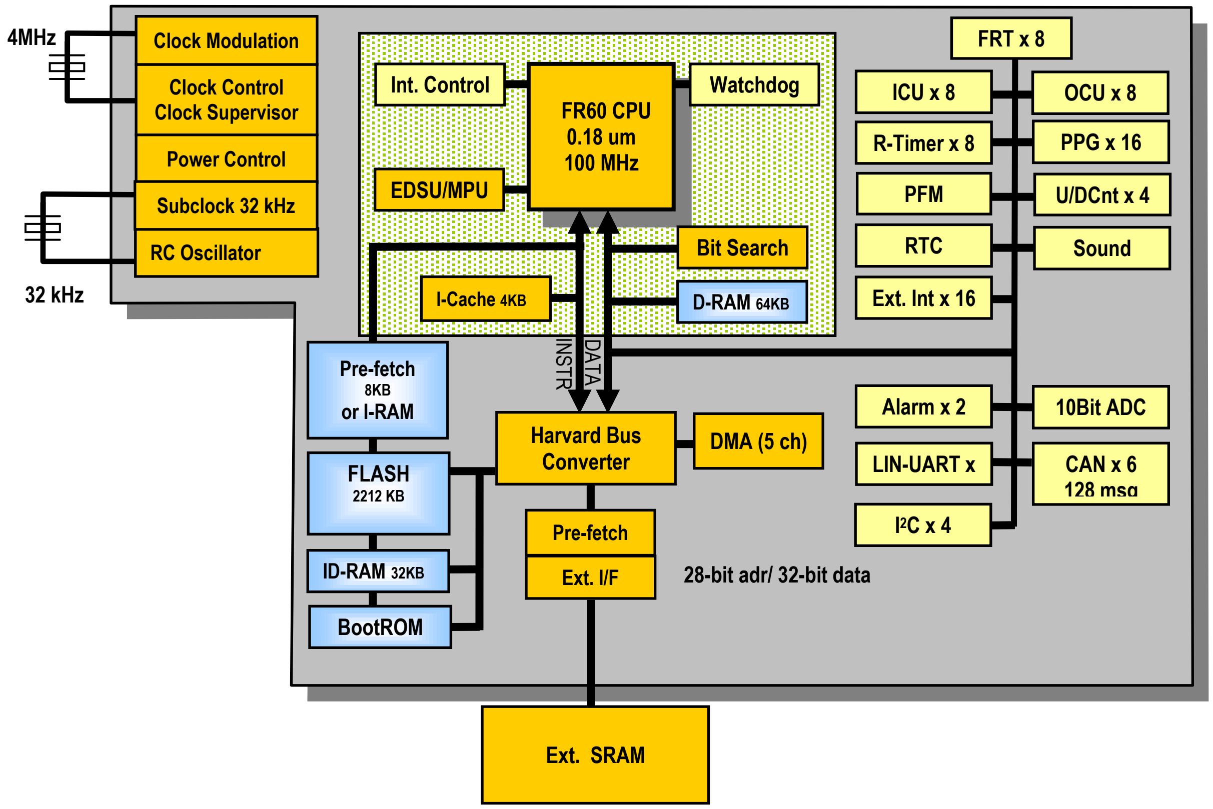
Contents

1 Introduction.....1
2 Sound Generation.....2
2.1 PCM (Pulse Code Modulated) Based Solutions..2
2.2 Tone Amplitude based solution.....17
Document History.....24
Worldwide Sales and Design Support.....25
Products.....25
PSoC® Solutions.....25
Cypress Developer Community.....25
Technical Support.....25

1 Introduction

This application note gives an overview off different sound generation possibilities on the MB91460 series MCU's.

Figure 1. MB91460 Series Block Diagram



The MB91460 series MCU's offers many different methods to output a sound.

This document shows some of these possibilities and gives some basic information's on how to set up these applications on an MB91460 series MCU based device.

2 Sound Generation

Sound generation is a comprehending term that is split into two different basic solutions for generation of sounds on embedded systems.

The first option is the PCM (Pulse code modulated) based sound generation. There are many different ways possible to transform a PCM based digital sound back into an audible tone.

The following list gives an overview for PCM based solutions on the MB91460 series MCU's:

- SG (Sound Generator) as 8Bit PWM (Puls Width Modulator)
- PPG (Puls Pause Generator) as 16Bit PWM
- DA (Digital-Analogue-Converter)

Some of these methods are feasible solutions. Each one has advantages and handicaps and has to be analyzed for each application. All these PWM solutions are based on a DMA data transfer to ensure an uninterruptible data stream.

The second solution is the tone amplitude generation based on the embedded Sound Generator.

- Sound Generator as Tone Amplitude Generator.

2.1 PCM (Pulse Code Modulated) Based Solutions

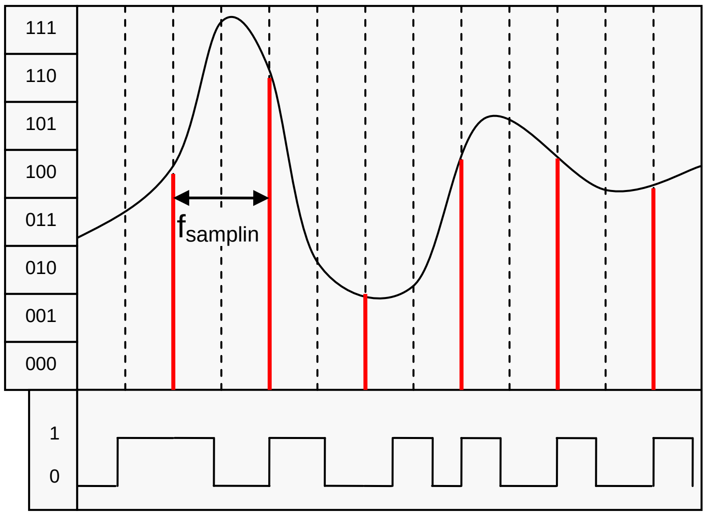
2.1.1 Basic PCM Knowledge

The PCM (Pulse code modulated) sound generation is bases on a simple Wavetable that holds the binary coded image of an analogue signal that is digitized with a predefined sample frequency.

Each lookup table value represents the amplitude height of the analogue sound signal at each sampling instance. The sampling frequency is a generally in the range from 16 kHz to 44.1 kHz. The sampling accuracy can be 8 or 16bit.

The following figure shows

Figure 2. PCM Wavetable Generation



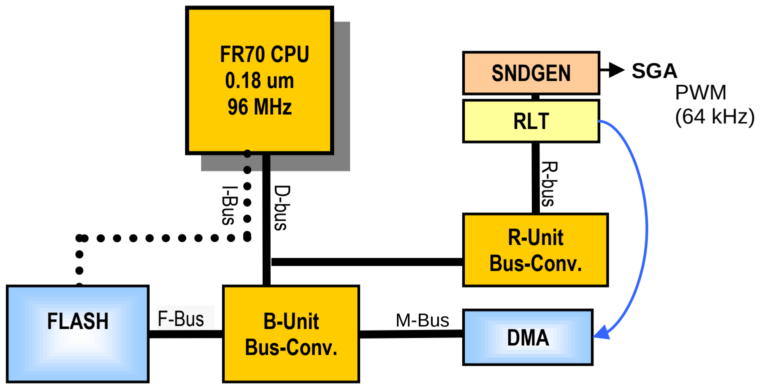
To reconstruct the analogue signal from the PCM Wavetable, each digitized sample value has to be output through an appropriate system with the defined sample frequency of this Wavetable. The suitable systems are mentioned in chapter 2 and are further illustrated in the following chapters.

2.1.2 Sound Generator as 8Bit PWM

The sound generation with the embedded Sound Generator as PWM in combination with DMA claims a high amount of resources within the MCU.

- Sound Generator
- Reload Timer
- DMA
- Flash memory
- Internal bus bandwidth

Figure 3. Included Resources in SG & DMA as PWM Sound Generation



Basic information’s

This Sound generation with the Sound Generator as a PWM is based upon a PCM (Pulse code modulated) Wavetable that holds the binary coded image of an analogue signal.

This PPG solution operates almost independent of the MCU core. This can be achieved by using the internal DMA controller for movement of the lookup table data from memory to the respective PPG registers.

After software triggering the reload timer, the DMA process is started and doesn’t need the MCU for any further operation. Except the DMA End Interrupt no MCU interaction is needed during playback.

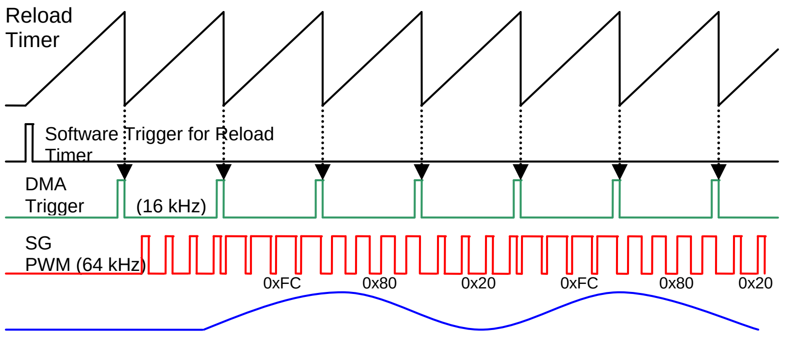
If the output Wavetable is larger than the maximum size of 64kByte, the MCU must reinitialize the DMA process to the higher Wavetable set and restart the DMA process.

If no sound playback is done, the SG duty cycle value is selected with 0%. To start the playback, the DMA source register has to be set to the first Wavetable entry in memory and the transfer count register holds the size of the Wavetable (NOTE: If the size of the Wavetable exceeds the 64kByte maximum transfer size of the DMA, an additional software solution within the DMA process is necessary).

After initialization of all resources the playback is started by triggering the reload timer. After this the whole playback process runs autonomous.

The playback process ends with execution of the DMA end interrupt service routine.

Figure 4. SG Wavetable Playback Signal Diagram



Setup for SG & DMA as PWM sound generation

Setup of Sound Generator

- SGCR (0x198)

S2	S1	S0	Clock input
0	0	0	CLK
0	0	1	1/2 CLK
0	1	0	1/4 CLK
0	1	1	1/8 CLK
1	0	0	1/16 CLK

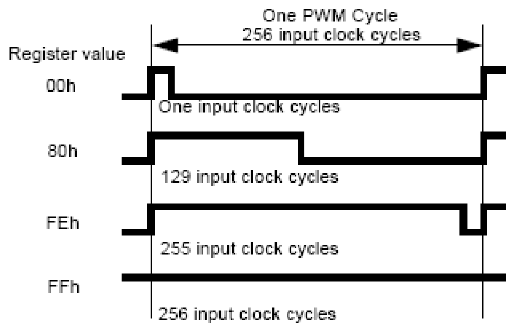
These bits specify the clock input signal for the Sound Generator.
S = 000 → Sound Generator Clock = CLKP

SGA PWM frequency = (CLKP / 256) * Prescaler = 16MHz/256 * 1/1 = 62.5kHz

- **SGAR (0x19C)**

The Amplitude Data register defines the duty cycle value for the SGA PWM signal.
One PWM Cycle is equal to 256 input clock cycles in every moment. The following figure shows the relationship between the register value and the PWM pulse.

SGAR = 0xFF = 100% duty cycle
SGAR = 0x00 = 2.56% duty cycle



This register is used for generating the amplitude modulation.

Setup of Reload Timer

■ **TMCSR0-7 (0x1B6 – 0x1EE)**

The control status register controls the operation mode of the reload timer and interrupts.

CSL2	CSL1	CSL0	Count clock
0	0	0	Internal clock CLKP/2
0	0	1	Internal clock CLKP/8
0	1	0	Internal clock CLKP/32
0	1	1	External event (external clock)
1	0	1	Internal clock CLKP/64
1	1	0	Internal clock CLKP/128

Count clock selection with CLKP = peripheral clock = 16MHz

CSL = 000 → Count Clock = 16MHz / 2 = 8 MHz

MOD2	MOD1	MOD0	Reload trigger
0	0	0	Software trigger
0	0	1	External trigger (rising edge)
0	1	0	External trigger (falling edge)
0	1	1	External trigger (both edges)

Reload trigger when internal clock is selected.

MOD = 000 → Software Trigger

RELD	Enable reload
0	One-shot mode (reload disabled)
1	Reload mode (reload enabled)

In reload mode, down counter underflow (0000H -> FFFFH) causes the value set to reload register (TMRLR) to be loaded to the down counter and the count operation continues.

RELD = 1 → automatic counter reload

INTE	Enable timer interrupt requests
0	Disable interrupt requests
1	Enable interrupt requests

When timer interrupt requests are enabled, the timer interrupt request flag (UF) becomes “1” and interrupt requests are generated.

INTE = 1 → generate Interrupt to trigger DMA process

- **TMRLR0-7 (0x1B0 – 0x1E8)**

  The reload value for the down counter is stored in reload register TMRLR. Please write using half-word access.

  Cycle = Count Clock (TMRLR + 1)

  62500ns (16kHz) = 125ns (8MHz) * (TMRLR + 1)

  TMRLR = 499 = 0x01F3

It is very important not to set the ICR value for the selected reload timer and not to set any interrupt service routine handler in ‘vectors.c’!

Setup of DMA channel

- DMACA0-4 (0x200 – 0x220)

  The control status register controls the operation mode of the reload timer and interrupts.

BLK	Function
$XXXX_B$	Block size of the corresponding channel

These bits specify the block size for block transfer on the corresponding channel. The value specified by these bits becomes the number of words in one transfer unit (more exactly, the repetition count of the data width setting).

BLK = 1

DTC	Function
$XXXX_B$	Transfer count for the corresponding channel

When DMA transfer is started, data in this register is stored in the counter buffer of the DMA-dedicated transfer counter and is decremented by 1 (subtraction) after each transfer unit. When DMA transfer is completed, the contents of the counter buffer are written back to this register and then DMA ends. This value represents the size of the selected Wavetable or Wavetable set. DTC = Wavetable size (maximum 0xFFFF → 64kByte Wavetable size)

IS	EIS	RN	Function	Transfer stop request
10010	0000	2	External Interrupt 2	-
10011	0000	3	External Interrupt 3	-
10100	0000	4	Reload Timer 0	-
10101	0000	5	Reload Timer 1	-
10110	0000	6	USART (LIN) 0 RX	available
10111	0000	7	USART (LIN) 0 TX	-

These bits select the source of a transfer request.
IS = 10100, EIS = 0000 → Reload Timer 0

- DMACB0-4 (0x204 – 0x224)

EDIE	Function
0	Disables end interrupt request output. (initial value)
1	Enables end interrupt request output.

This bit controls the occurrence of an interrupt for normal termination.
EDIE = 1, enable Interrupt after the transfer of the Wavetable is complete.

SASZ	Function
XX_H	Specify the increment/decrement width of the transfer source address. 0 to 255

These bits specify the increment or decrement width for the transfer source address (DMASA) of the corresponding channel in each transfer operation. The value set by these bits becomes the address increment/decrement for each transfer unit. This value is stated in byte width.

WS	Function
00_B	Byte-width transfer (initial value)
01_B	Halfword-width transfer
10_B	Word-width transfer
11_B	Setting disabled

These bits are the transfer data width selection bits and are used to select the transfer data width of the corresponding channel. Transfer operations are repeated in units of the data width specified in this register for as many times as the specified count.

WS = 00 , Byte → 8Bit
WS * BLK = 8Bit * 1 = 8Bit / Transfer

- DTC * WS * BLK = Wavetable size e.g. (1024) * 8Bit * 1 = 8192 Bit overall transfer
- Reload Timer 0 → 16kHz (62.5us)

1024 * 62.5us = 64ms duration of generated sound with 8bit sample accuracy.

- DMASA0-4 (0x1000 – 0x1020)

  These bits set the transfer source address.
  DMASAx = Wavetable pointer, this register holds the pointer to the first Wavetable entry.

- DMADA0-4 (0x1004 – 0x1024)

  These bits set the transfer destination address.
  DMADAx = &SGAR, destination is the duty cycle register of the Sound Generator.

Handicap

- Bus priority on internal bus.

  If the MCU claims the internal D-Bus or R-Bus for large or iterating data transfer to resources, D-RAM or any other destination, the DMA block transfer from Flash to the SG resource is immediately stopped, because the MCU core has the highest priority on the internal bus system. In certain conditions this could lead to a timing failure in the selected sample rate and thus to a worse sound playback with clocks and distortion.

  Calculation of this behavior is very difficult and highly depending on the respective application.

Advantage

- 8 Bit access to SG registers.

  The access to the SG duty cycle register is possible in byte mode.
  This is leading to half sized Wavetable than it is needed for a solution with the Programmable Pulse Generator.

- Uncomplicated conversion of standardized sound formats e.g. WAV or MP3 to PCM Wavetable files for inclusion into this SG sound generation.

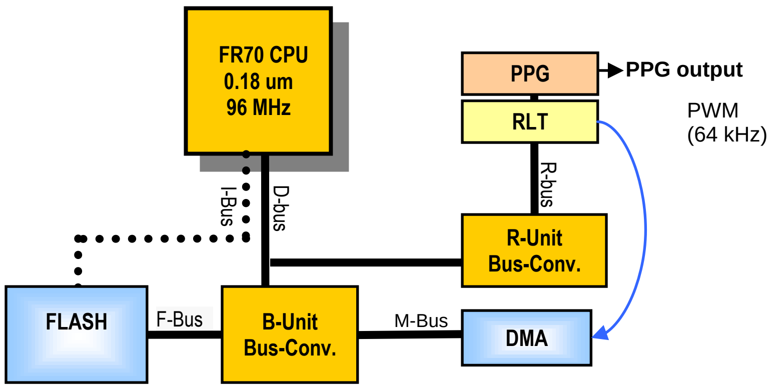
- Almost no MCU core interruption during sound playback

2.1.3 Programmable Pulse Generator as 16Bit PMW

The sound generation solution with the Programmable Pulse Generator as PWM in combination with DMA claims the same amount of resources within the MCU as the equivalent Sound Generator solution.

- Programmable Pulse Generator
- Reload Timer
- DMA
- Flash memory
- Internal bus bandwidth

Figure 5. Included Resources in PPG Sound Generation



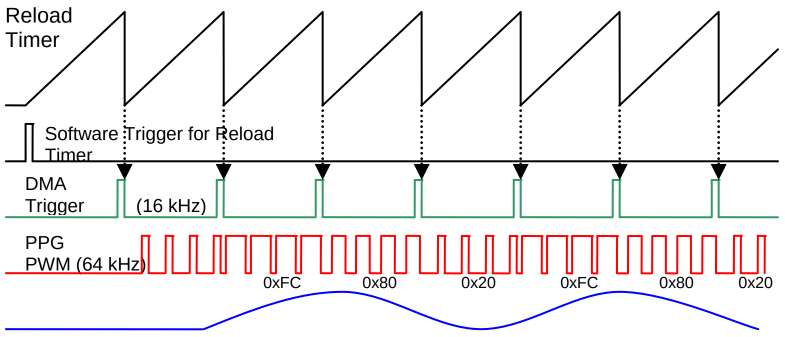
Basic information's

This Sound generation with the Programmable Pulse Generator is based upon a PCM (Pulse code modulated) Wavetable that holds the binary coded image of an analogue signal which was previously digitized with a predefined sample frequency and accuracy.

This PPG solution operates almost independent of the MCU core. After software triggering the reload timer, the DMA process is started and doesn't need the MCU for any further operation. Except the DMA End Interrupt no MCU interaction is needed during the playback.

If the output Wavetable is larger than the maximum size of 128kByte, the MCU must reinitialize the DMA process to the higher Wavetable set and restart the DMA process. If no sound playback is done, the PPG duty cycle value is selected with 0%. To start the playback, the DMA source register has to be set to the first Wavetable entry in memory and the transfer count register holds the size of the Wavetable (NOTE: If the size of the Wavetable exceeds the 128kByte maximum transfer size of the DMA, an additional software solution within the DMA process is necessary). After initialization of all resources the playback is started by triggering the reload timer. After this the whole playback process runs autonomous.
The playback process ends with execution of the DMA end interrupt service routine.

Figure 6. PPG Wavetable Playback Signal Diagram



Setup for PPG sound generation

RLT Register Settings

- TMCSR0-7 (0x1B6 – 0x1EE)

  The control status register controls the operation mode of the reload timer and interrupts.

CSL2	CSL1	CSL0	Count clock
0	0	0	Internal clock CLKP/2
0	0	1	Internal clock CLKP/8
0	1	0	Internal clock CLKP/32
0	1	1	External event (external clock)
1	0	1	Internal clock CLKP/64
1	1	0	Internal clock CLKP/128

  Count clock selection with CLKP = peripheral clock = 16MHz

  CSL = 000 → Count Clock = 16MHz / 2 = 8 MHz

MOD2	MOD1	MOD0	Reload trigger
0	0	0	Software trigger
0	0	1	External trigger (rising edge)
0	1	0	External trigger (falling edge)
0	1	1	External trigger (both edges)

  Reload trigger when internal clock is selected.

  MOD = 000 → Software Trigger

RELD	Enable reload
0	One-shot mode (reload disabled)
1	Reload mode (reload enabled)

  In reload mode, down counter underflow (0000H -> FFFFH) causes the value set to reload register (TMRLR) to be loaded to the down counter and the count operation continues.

RELD = 1 → automatic counter reload

INTE	Enable timer interrupt requests
0	Disable interrupt requests
1	Enable interrupt requests

When timer interrupt requests are enabled, the timer interrupt request flag (UF) becomes “1” and interrupt requests are generated.

INTE = 1 → generate Interrupt to trigger DMA process

- TMRLR0-7 (0x1B0 – 0x1E8)

  The reload value for the down counter is stored in reload register TMRLR. Please write using half-word access.

  Cycle = Count Clock (TMRLR + 1)

  62500ns (16kHz) = 125ns (8MHz) * (TMRLR + 1)

  TMRLR = 499 = 0x01F3

  It is very important not to set the ICR value for the selected reload timer and not to set any interrupt service routine handler in ‘vectors.c’!

DMA register settings

- **DMACA0-4 (0x200 – 0x220)**

  The control status register controls the operation mode of the reload timer and interrupts.

BLK	Function
$XXXX_B$	Block size of the corresponding channel

These bits specify the block size for block transfer on the corresponding channel. The value specified by these bits becomes the number of words in one transfer unit (more exactly, the repetition count of the data width setting).

BLK = 1

DTC	Function
$XXXX_B$	Transfer count for the corresponding channel

When DMA transfer is started, data in this register is stored in the counter buffer of the DMA-dedicated transfer counter and is decremented by 1 (subtraction) after each transfer unit. When DMA transfer is completed, the contents of the counter buffer are written back to this register and then DMA ends. This value represents the size of the selected Wavetable or Wavetable set. DTC = Wavetable size (maximum 0xFFFF → 128kByte Wavetable size)

IS	EIS	RN	Function	Transfer stop request
10010	0000	2	External Interrupt 2	-
10011	0000	3	External Interrupt 3	-
10100	0000	4	Reload Timer 0	-
10101	0000	5	Reload Timer 1	-
10110	0000	6	USART (LIN) 0 RX	available
10111	0000	7	USART (LIN) 0 TX	-

These bits select the source of a transfer request.

IS = 10100, EIS = 0000 → Reload Timer 0

- DMACB0-4 (0x204 – 0x224)

EDIE	Function
0	Disables end interrupt request output. (initial value)
1	Enables end interrupt request output.

This bit controls the occurrence of an interrupt for normal termination.
EDIE = 1, enable Interrupt after the transfer of the Wavetable is complete.

SASZ	Function
XX_H	Specify the increment/decrement width of the transfer source address. 0 to 255

These bits specify the increment or decrement width for the transfer source address (DMASA) of the corresponding channel in each transfer operation. The value set by these bits becomes the address increment/decrement for each transfer unit. This value is stated in byte width.

WS	Function
00_B	Byte-width transfer (initial value)
01_B	Halfword-width transfer
10_B	Word-width transfer
11_B	Setting disabled

These bits are the transfer data width selection bits and are used to select the transfer data width of the corresponding channel. Transfer operations are repeated in units of the data width specified in this register for as many times as the specified count.

WS = 01 , Byte → 16Bit
WS * BLK = 16Bit * 1 = 16Bit / Transfer

- DTC * WS * BLK = Wavetable size e.g. (1024) * 16Bit * 1 = 16384 Bit overall transfer
- Reload Timer 0 → 16kHz (62.5us)

1024 * 62.5us = 64ms duration of generated sound with 16 bit sample accuracy.

- DMASA0-4 (0x1000 – 0x1020)

These bits set the transfer source address.
DMASAx = Wavetable pointer, this register holds the pointer to the first Wavetable entry.

- **DMADA0-4 (0x1004 – 0x1024)**

These bits set the transfer destination address.
DMADAx = &PDUTy, destination is the duty cycle register of the Programmable Pulse Generator.

PPG register settings

- **PCN0-15 (0x116 – 0x34E)**

CKS1	CKS0	Down Counter Count Clock Selection
0	0	Peripheral clock (CLKP)
0	1	Peripheral clock divided by 4
1	0	Peripheral clock divided by 16
1	1	Peripheral clock divided by 64

Counter clock selection. CKS = 00 → Counter Clock = CLKP = 16MHz

RTTG	Operation
0	Disable restart.
1	Enable restart.

- When the Enable Restart bit is set to "1", a trigger (software/internal) is generated to enable a restart.

  RTTG = 1

- **PCSR0-15 (0x112 – 0x34A)**
  Controls the cycle of the PPG.

  Cycle = Count Clock (PCSR + 1)

  15625ns (64kHz) = 62.5ns (16MHz) (PCSR +1)

  PCSR = 249

Handicap

- 16 Bit access to PPG registers.

  The access to the PPG duty cycle register is only possible in half-word mode.
  This is leading to double sized Wavetable.

  Cycle = Count Clock (PCSRx + 1)
  15625ns (64kHz) = 62.5ns (16MHz) (PCSRx +1)

  PCSRx = 249 = 0x00F9

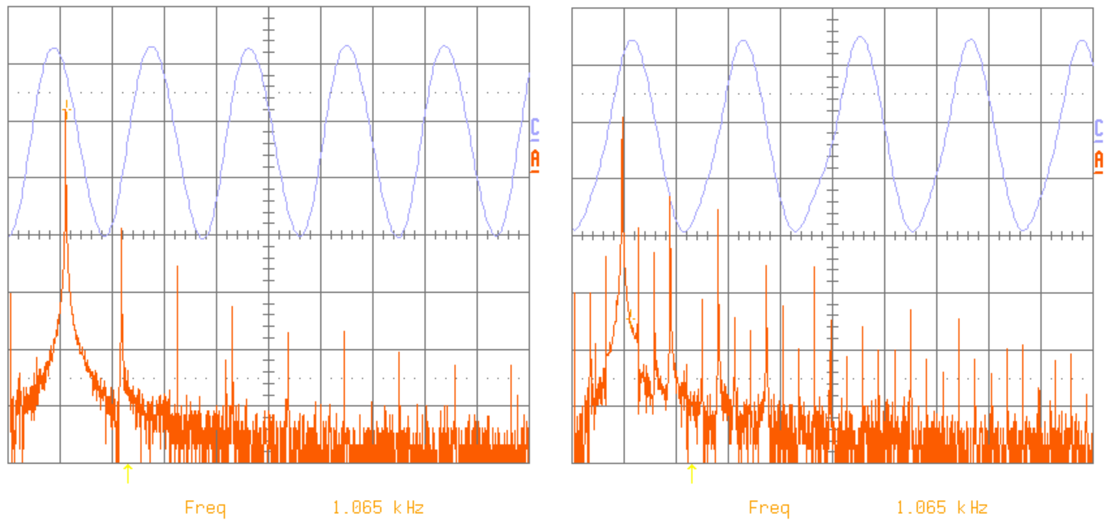
  The setting of 0x00F9 leads to a maximum (100%) duty cycle setting of
  PDUTx = 0x00F9.

  All values in the Wavetable have to be 16Bit although they are only necessary in 8Bit width. This leads to double sized Wavetable!

- Bus priority on internal bus.

  If the MCU claims the internal D-Bus or R-Bus for large or iterating data transfer to resources, D-RAM or any other destination, the DMA block transfer from Flash to the PPG resource is immediately stopped, because the MCU core has the highest priority on the internal bus system. In certain conditions this could lead to a timing failure in the selected sample rate and thus to a worse sound playback with clocks and distortion.

  Calculation of this behavior is very difficult and highly depending on the respective application.



The figures above show a unaffected sound generation (left figure) of a 1kHz sine signal (blue) and the appropriate FFT (red), that shows some additional noise which is based on the power supply.

The second figure (right) shows the intrusion of a high frequency access of the MCU to some resources connected to the R-Bus. This causes very high interference on the sound generation and produces wideband noise at the output.

Advantages

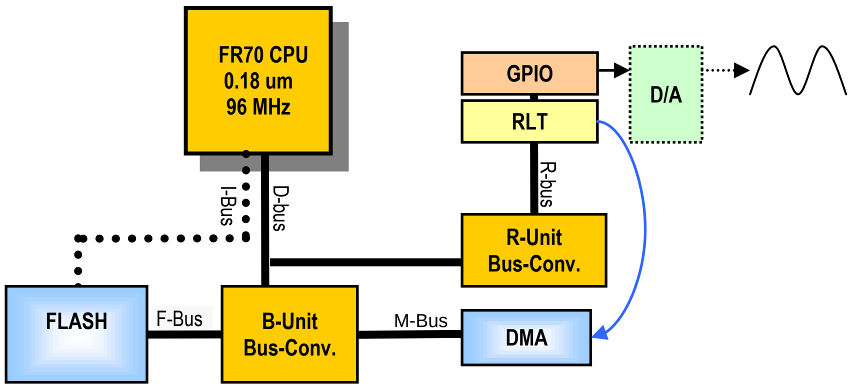
- Uncomplicated conversion of standardized sound formats e.g. WAV or MP3 to PCM Wavetable files for inclusion into the PPG sound application.
- Almost no MCU core interruption during sound playback

2.1.4 8Bit Digital-Analogue Converter

The sound generation with a Digital-Analogue Converter in combination with DMA claims, like all DMA solutions a high amount of resources within the MCU and an additional external resource for digital to analogue conversion.

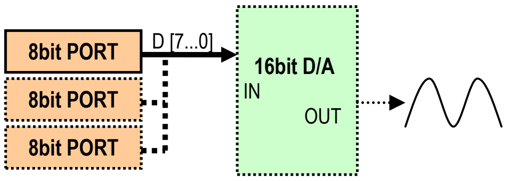
- General Purpose IO
- Reload Timer
- DMA
- Ext. D/A converter
- Flash memory
- Internal bus bandwidth

Figure 7. Included Resources in Digital-Analogue Converter for Sound Generation



Basic information's

This Sound generation with a Digital-Analogue Converter is based upon a PCM (Pulse code modulated) Wavetable that holds the binary coded image of an analogue signal. These values are sent to a D/A converter through simple General Purpose IO's. The bit depth and thus the count of used GPIO resources depend on the accuracy of the external connected D/A converter.



The size of the memorized Wavetable is also proportional to the selected accuracy of the D/A converter. But this solution gives the highest variability in sample rates, accuracy and thus memory consumption.

- Setup for GPIO & DMA for sound generation
- Setup of General Purpose IO's

- **PFRx (0xD80 – 0xD9D)**

PFRx.y	0 – Port pin y is in general purpose mode
	1 – Port pin y is in resource function

The selected output port is set to general purpose IO functionality to feed the D/A converter input.

PFRx = 0x00

- **DDRx (0xD40 – 0xD5D)**

DDRx.y	0 – Port pin y is in input mode
	1 – Port pin y is in output mode

The selected output port is set to output functionality to feed the D/A converter input.

DDRx = 0xFF

- **PODRx (0xE00 – 0xE1D)**

Bit	Port Output Drive Registers	
	0 (default)	1
PODRx.y	5 mA output drive	2 mA output drive

Almost all port feature a programmable output drive strength selection for each port pin.

The default value for this is 5mA!

Setup of Reload Timer

- **TMCSR0-7 (0x1B6 – 0x1EE)**
The control status register controls the operation mode of the reload timer and interrupts.

CSL2	CSL1	CSL0	Count clock
0	0	0	Internal clock CLKP/2
0	0	1	Internal clock CLKP/8
0	1	0	Internal clock CLKP/32
0	1	1	External event (external clock)
1	0	1	Internal clock CLKP/64
1	1	0	Internal clock CLKP/128

Count clock selection with CLKP = peripheral clock = 16MHz

CSL = 000 → Count Clock = 16MHz / 2 = 8 MHz

MOD2	MOD1	MOD0	Reload trigger
0	0	0	Software trigger
0	0	1	External trigger (rising edge)
0	1	0	External trigger (falling edge)
0	1	1	External trigger (both edges)

Reload trigger when internal clock is selected.

MOD = 000 → Software Trigger

RELD	Enable reload
0	One-shot mode (reload disabled)
1	Reload mode (reload enabled)

In reload mode, down counter underflow (0000H -> FFFFH) causes the value set to reload register (TMRLR) to be loaded to the down counter and the count operation continues.

RELD = 1 → automatic counter reload

INTE	Enable timer interrupt requests
0	Disable interrupt requests
1	Enable interrupt requests

When timer interrupt requests are enabled, the timer interrupt request flag (UF) becomes “1” and interrupt requests are generated.

INTE = 1 → generate Interrupt to trigger DMA process

- **TMRLR0-7 (0x1B0 – 0x1E8)**
  The reload value for the down counter is stored in reload register TMRLR. Please write using half-word access.

  Cycle = Count Clock (TMRLR + 1)

  62500ns (16kHz) = 125ns (8MHz) * (TMRLR + 1)

  TMRLR = 499 = 0x01F3

  This value represents the sample rate!

It is very important not to set the ICR value for the selected reload timer and not to set any interrupt service routine handler in 'vectors.c'!

Setup of DMA channel

- **DMACA0-4 (0x200 – 0x220)**
  The control status register controls the operation mode of the reload timer and interrupts.

BLK	Function
$XXXX_B$	Block size of the corresponding channel

These bits specify the block size for block transfer on the corresponding channel. The value specified by these bits becomes the number of words in one transfer unit (more exactly, the repetition count of the data width setting).

BLK = 1

DTC	Function
$XXXX_B$	Transfer count for the corresponding channel

When DMA transfer is started, data in this register is stored in the counter buffer of the DMA-dedicated transfer counter and is decremented by 1 (subtraction) after each transfer unit. When DMA transfer is completed, the contents of the counter buffer are written back to this register and then DMA ends. This value represents the size of the selected Wavetable or Wavetable set. DTC = Wavetable size (maximum 0xFFFF → 128kByte Wavetable size)

IS	EIS	RN	Function	Transfer stop request
10010	0000	2	External Interrupt 2	-
10011	0000	3	External Interrupt 3	-
10100	0000	4	Reload Timer 0	-
10101	0000	5	Reload Timer 1	-
10110	0000	6	USART (LIN) 0 RX	available
10111	0000	7	USART (LIN) 0 TX	-

These bits select the source of a transfer request.
IS = 10100, EIS = 0000 → Reload Timer 0

- **DMACB0-4 (0x204 – 0x224)**

EDIE	Function
0	Disables end interrupt request output. (initial value)
1	Enables end interrupt request output.

This bit controls the occurrence of an interrupt for normal termination.
EDIE = 1, enable Interrupt after the transfer of the Wavetable is complete.

SASZ	Function
XX_H	Specify the increment/decrement width of the transfer source address. 0 to 255

These bits specify the increment or decrement width for the transfer source address (DMASA) of the corresponding channel in each transfer operation. The value set by these bits becomes the address increment/decrement for each transfer unit. This value is stated in byte width.

WS	Function
00_B	Byte-width transfer (initial value)
01_B	Halfword-width transfer
10_B	Word-width transfer
11_B	Setting disabled

These bits are the transfer data width selection bits and are used to select the transfer data width of the corresponding channel. Transfer operations are repeated in units of the data width specified in this register for as many times as the specified count.

This value highly depends on the selected external D/A converter input accuracy.
In this example the input accuracy is 8bit!

WS = 00 , Byte → 8Bit
WS * BLK = 8Bit * 1 = 8Bit / Transfer

- DTC * WS * BLK = Wavetable size e.g. (1024) * 8Bit * 1 = 8192 Bit overall transfer
- Reload Timer 0 → 16 kHz (62.5us)

  1024 * 62.5us = 64ms duration of generated sound with 8 bit sample accuracy.

- **DMASA0-4 (0x1000 – 0x1020)**
  These bits set the transfer source address.
  DMASAx = Wavetable pointer, this register holds the pointer to the first Wavetable entry.

- **DMADA0-4 (0x1004 – 0x1024)**
  These bits set the transfer destination address.
  DMADAx = &PDRx, destination is the duty cycle register of the selected general purpose IO port.

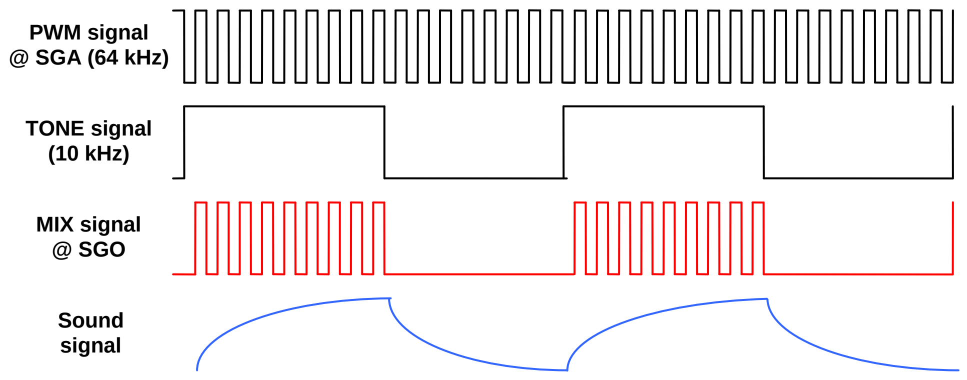
Handicap

Advantage

2.2 Tone Amplitude based solution

The Tone Amplitude modulation is generally based on two different mixed signals. The first of these two signals has a fixed clock and a variable pulse width for generation of the amplitude. The second signal represents the variable tone frequency with a fixed pulse width.

Figure 8. Basic Tone Amplitude Output Signal



2.2.1 Sound Generator

Figure 9. Included Resources in Sound Generator Playback

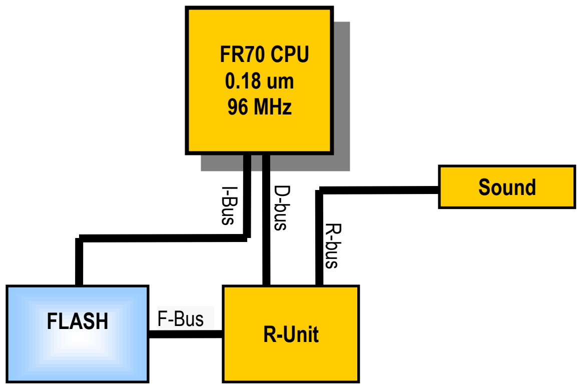
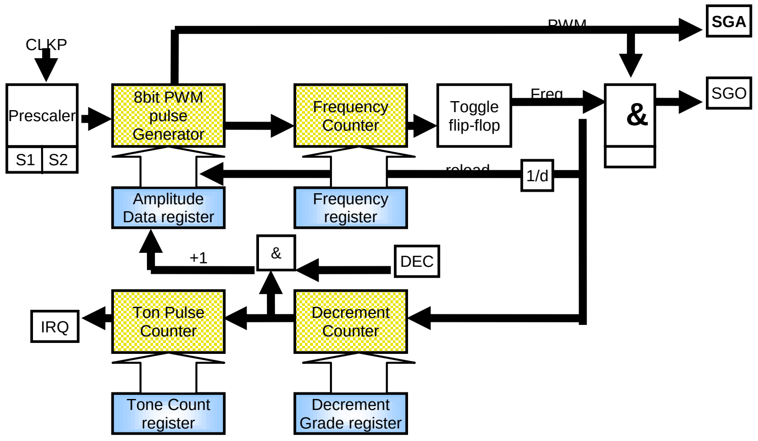


Figure 10. Sound Generator Block Diagram

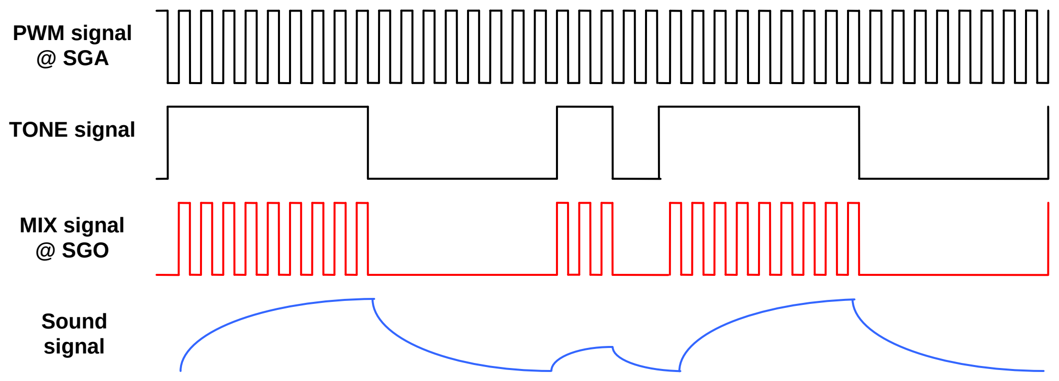


Basic Information’s

The Sound Generator is generally a Pulse Width Modulator that mixes to different PWM signals to generate a tone at the output. The duty cycle of these two PWM signals define the tone frequency and the amplitude of the generated sound.

The first PWM signal is the TONE signal. It has a fixed duty cycle of 50% and an adjustable cycle frequency. The adjustable cycle frequency of this signal is equally set for each generated sound.

The second PWM signal has a variable duty cycle and a preset able cycle frequency. The cycle frequency of this signal is defined by the Resource Clock and an internal and adjustable Prescaler. This frequency is set during initialization of the sound generator. The duty cycle of this signal represents the amplitude of the generated sound and thus it is set for each generated sound respectively.



The generated sound signal is not comparable to a real sine wave, because the Sound Generator isn’t able to vary the PWM duty cycle of the SGA signal within one tone cycle. Thus the Sound Generator is not able to produce speech or similar music.

The cycle frequency of the SGA PWM signal is preset during initialization of the Sound Generator and stays at this value for the whole operation. The duty cycle of this PWM signal is variable to represent the amplitude of the generated sound signal.

The cycle frequency of the TONE signal poses the generated sound frequency. This signal has a fixed duty cycle value to create the analogy to a sine wave.

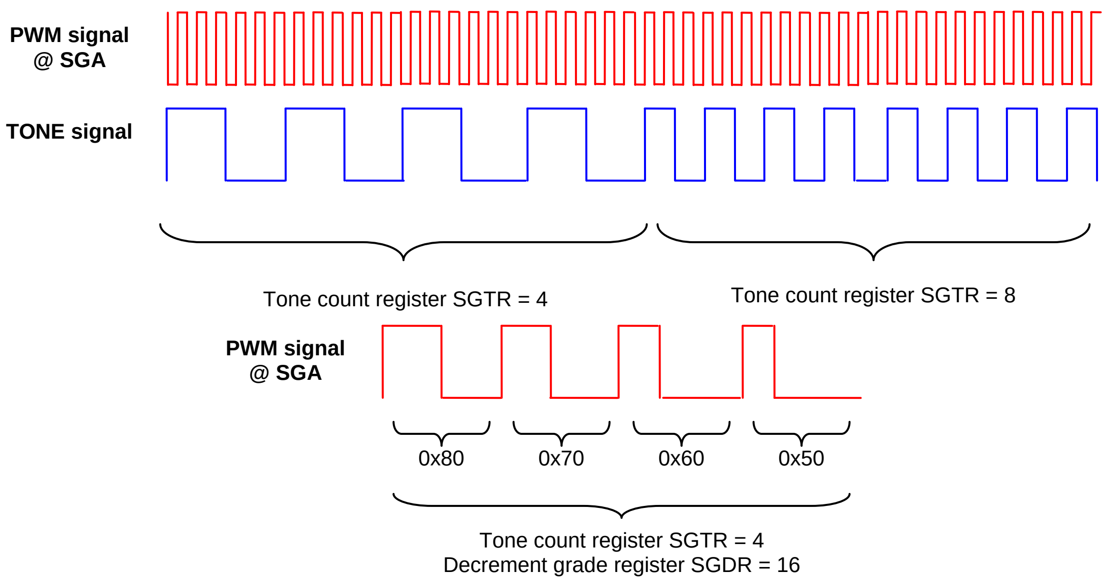
The Sound Generator solution is based on a frequency, amplitude and decay grade for each generated tone.

```
typedef struct {
       char freq;
       char ampl;
       char decay;
} stSndItem;
const stSndItem SndRocknRoll[] = {
       {30,0,0},    /* melody tone count reg. value */
       {59,30,4}, /* frequency, amplitude, decay */
       {47,40,2},
       {39,64,1},
       {35,64,2},
       {33,64,1},
            …
}
```

The melody tone count defines the duration of each played tone. The Sound Generator has an internal tone counter, with a respective count register SGTR, which counts each generated tone cycle. If the internal counter reaches the preset reload value stored in the count register an interrupt is generated.

Within the interrupt service code the values for the next tone are loaded into the tone, amplitude and decay registers of the sound generator.

The decay grade value implies the decrement step for the amplitude value SGAR for each tone cycle and consequently it represents the amplitude reduction for each tone cycle.


PWM signal @ SGA
TONE signal
Tone count register SGTR = 4
Tone count register SGTR = 8
PWM signal @ SGA
0x80
0x70
0x60
0x50
Tone count register SGTR = 4
Decrement grade register SGDR = 16

The size of these tables in memory is hardly depending on the played melody itself. Thus a value like *Byte/sec*, whereby *sec* implies one second of sound playback, is not calculable because each melody and consequently each note has a different value for tone count and decrement grade. These values are generally depending on the BPM value for this respective melody.

The following table gives an example on the duration of notes for a BPM value of 60:

Note	Duration [ms]
Dot 1/2 note:	3000
1/2 note:	2000
Dot 1/4 note:	1500
1/4 note:	1000
Dt 1/8 note:	750
¼T note:	666.667

Note	Duration [ms]
1/8 note:	500
Dot 1/16 note :	375
1/8T note:	333.333
1/16 note:	250
1/16T note:	166.667
1/32 note:	125

To roughly calculate the size of Tone-Amplitude table for respectively one special melody with no reference to the duration of the played melody, the following formula is applicable:

Nbr_{note} * 3 Byte = table size [Byte]

$Nbr_{note} \rightarrow$ number of notes in melody

Because of the rare accesses to the Sound Generator registers, only three register accesses per tone, the used bandwidth on internal busses like D-Bus, R-Bus or F-Bus is very low. Although there is always one interrupt necessary for each note that is played.

Wrong settings for the tone count register SGTR can trigger a high busload on internal busses. This additionally forces a very high interrupt load for the MCU, because a low setting for the tone count register or a high value for the decrement grade register SGDR might lead to a very high interrupt frequency or actually one interrupt for each tone cycle.

Calculating the exact bandwidth usage on R-, D- and F-Bus is very difficult.

Setup for Sound Generator playback

Sound Generator register settings

- **SGCR (0x198)**

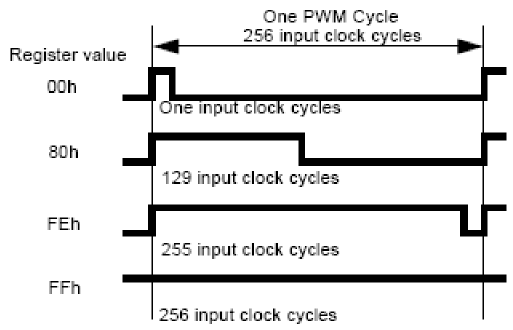
S2	S1	S0	Clock input
0	0	0	CLK
0	0	1	1/2 CLK
0	1	0	1/4 CLK
0	1	1	1/8 CLK
1	0	0	1/16 CLK

These bits specify the clock input signal for the Sound Generator.
S = 000 → Sound Generator Clock = CLKP

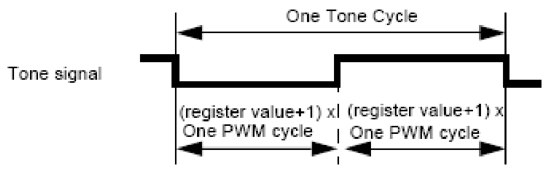
SGA PWM frequency = (CLKP / 256) * Prescaler = 16MHz/256 * 1/1

- **SGAR (0x19C)**
The Amplitude Data register defines the duty cycle value for the SGA PWM signal.
One PWM Cycle is equal to 256 input clock cycles in every moment. The following figure shows the relationship between the register value and the PWM pulse.

SGAR = 0xFF = 100% duty cycle
SGAR = 0x00 = 2.56% duty cycle



- **SGFR (0x19A)**
The Frequency data register defines the tone frequency of the generated sound signal.
TONE frequency = (CLKP / 256) * S2-S1-S0-Prescaler * SGFR * 2


One Tone Cycle
Tone signal
(register value+1) x One PWM cycle
(register value+1) x One PWM cycle

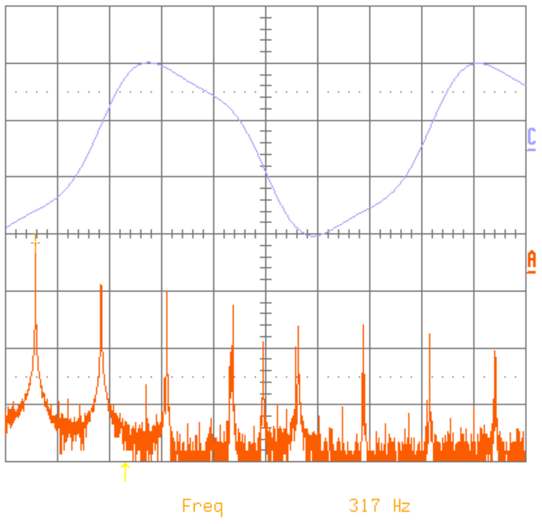
Handicap

- Worse sound quality.

The generated sound is not comparable to the sound playback generated by PPG solution.

- Distorted low frequency output signal

  Because of the long PWM output signal low phase the produced analogue signal is not comparable to a real sine wave and thus it produces a distorted signal at the output.



The above figure shows the distorted sine wave and the respective FFT, generated by the Sound Generator.

Advantages

- Low interrupt charge.

  The Sound Generator registers Amplitude Data, Frequency Data and the Decrement Grade have to set after each Tone has been played. This can only be done by software. But the interrupt frequency can be manually adjusted by changing the Tone count register.

- No influence on high Bus load by MCU accesses.

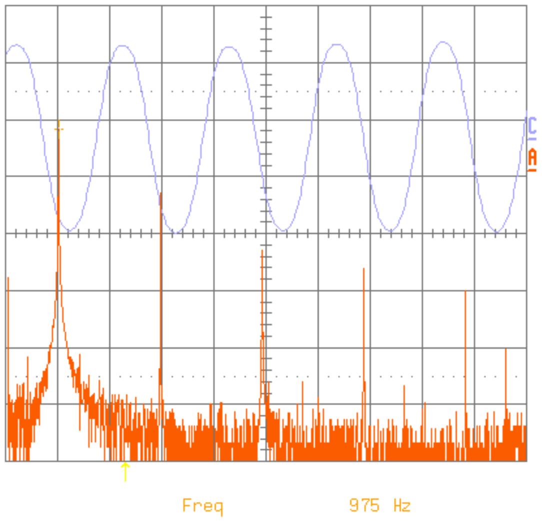
  The Sound Generated is not vulnerable to high internal bus loads, because the frequency of internal bus accesses

- Small memory image.

  The melody table memory images are very small, because there are only 96bit/second tone information’s necessary.

- Comparable signal quality for sine wave signals

  Sine wave signals above 1 kHz, generated by the Sound Generator, have a quality which is comparable to the sound quality generated by the PPG solution.



The above figure shows the 1 kHz sine wave and the respective FFT (red) generated by the Sound Generator.

Document History

Document Title: AN205204 - FR Family MB91460 Series, Sound Generation

Document Number: 002-05204

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	NOFL	03/14/2006	Initial release
*A	5085600	NOFL	01/14/2016	Migrated Spansion Application Note from MCU-AN-300027-E-V10 to Cypress format
*B	5873436	AESATMP8	09/05/2017	Updated logo and Copyright.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709