Nuvoton ChipCorder

One-Pin triggering Control Implementation Guide

- Use only one GPIO pin to control a ChipCorder device

Important Notice

Nuvoton products are not designed, intended, authorized or warranted for use as components in systems or equipment intended for surgical implantation, atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, or for other applications intended to support or sustain life. Furthermore, Nuvoton products are not intended for applications wherein failure of Nuvoton products could result in or lead to a situation wherein personal injury, death or severe property or environmental damage could occur.

Nuvoton customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nuvoton for any damages resulting from such improper use or sales.

1. Introduction

This documents aims to describe how to use only one GPIO pin to control a Nuvoton ChipCorder device. That is, by utilizing the Nuvoton ChipCorder's special features such as Voice Macro, GPIO triggering and Fast-Debounce, with a single GPIO pin a simple yet powerful and flexible control can be achieved so that the master MCU can select to play a random Voice Macro (VM) or Voice Prompt (VP), by sending a sequence of pulses following the timing specification defined in this document.

This document applies on ISD15D00, ISD3800, ISD2100 and ISD2360 series devices.

2. Implementation

The one-pin control of a Nuvoton ISD ChipCorder device is achieved by the cooperation of two parties:

1) the Voice Macro time machine running inside of the device after a trigger event occurs,
2) a sequence of GPIO triggers which is controlled by the master MCU.

The two parties both need to run or control the event on one common time line. The underlying idea is to let the trigger happen at the right time when device is waiting for possible execution branch.

Once triggered, the device internal Voice Macro machine starts to run: fetch and then execute the VM script command data from the flash memory. During the execution, it has to pause and give a time window for detecting if there is more trigger is yet to come. If another trigger is detected in this window, then the device will branch to execute another VM which is associated with this new trigger – thanks to the GPIO trigger's preemptive execution nature; otherwise the device will stop the waiting and resume the execution and run to the end.

On the other hand, the master MCU needs to generate the triggering edge on the IO line inside the VM waiting window – following the spec which is defined in section 2.2, so that the two parties can work together and ensure the desired VM get triggered and executed.

The following sections further describe the implementation details.

2.1 GPIO Trigger and Voice Macro

- Fast de-bounce

The one-pin control solution is only recommended for device which has fast de-bounce feature, so that the good system responsiveness can be achieved. After reset, by default the fast de-bounce is disabled and the de-bounce time is 20ms. Writing one to the fast de-bounce bit enables the fast de-bounce feature which requires only 8ns signal holding time. Refer to table 2-1 for fast de-bounce bit control configuration.

The fast de-bounce requires clean triggering signal. Usually for fast de-bounce the triggering signal level transition should only be controlled by on board MCU through an IO port, not by button pressing.

ISD2100	Register 0x17 bit5 [- - x - - - - -]	Default: 0x81. 1: enable. 0: disable
ISD2360	Register 0x17 bit5 [- - x - - - - -]	Default: 0x00 1: enable. 0: disable
ISD15D00, ISD3800	Register 0x30 bit2 [- - - - - x - -]	Default: 0xD1 1: enable. 0: disable

Table 2-1 Fast De-bounce control configuration

- Falling edge trigger

To simplify the implementation, this guide uses falling edge trigger. The sample VPE project distributed along with this document also only uses the falling edge trigger.

- Wakeup VM and GPIO associated VM

Once a triggering event is detected in PD state, the device will first execute the Wakeup VM, and then execute the VM associated with the triggering pin. If the GPIO trigger happens in PU state, then the device will skip the Wakeup VM and only execute the GPIO associated VM.

The implementation here assumes that the very first initial trigger always happens in PD state, and only uses the GPIO0 pin to generate the trigger. The index for GPIO0 associated VM is stored in R0. So after the first trigger, the device will always execute Wakeup VM and then the VM@R0, see Voice Macro Execution Flow Chat in Figure 2-1.

- GPIO trigger preemption

Voice Macro's execution is preemptive. That is, after triggered and during the VM execution, if another trigger event occurs, the device will stop the current VM execution and start to execute the VM associated with the new trigger.

Because of this preemptive feature, it made possible that a sequence of triggering events happened on the same IO pin can let device jumping through VMs and eventually landing on the desired VM and only let that VM execute to finish.

The trick is after the trigger, the command script changes the index stored in R0; so next trigger on the very same IO line will cause the VM pointed by new index preempt the current VM; and so on. Thus a chain of VM execution can be established. For detailed Voice Macro execution flow, please refer to figure 2-1.

- GPIO Trigger Timing Control

The critical part is inside of each VM a window frame should be given and the possible jumping trigger, which moves the execution along the chain, can only happen in this window. Inside that window, the device pauses VM execution, stays idle waiting for trigger. Once this waiting period is over, the device will resume the execution and play the real sound effect associated with that trigger.

This way actually changes how the VM is associated. Instead of the association between the VM and the physical GPIO pin, the new association is established between VM and the number of triggers. In other words, to let the device play the right sound, the on board MCU should issue the right amount of triggers, or pulses. The timing of trigger, and also the pulse's width will be further explained below.

2.1.1 Voice Macro Description

- Voice Macro Execution Flow

Figure 2-1 shows the one-pin control Voice Macro execution flow. Please note that the first VM after Wakeup VM should be used for common initialization; and all the following VMs can be configured as task VMs, which means besides the index moving, the later part of the VM can do the real playback task. For more details on VM configuration, please refer to the sample project.

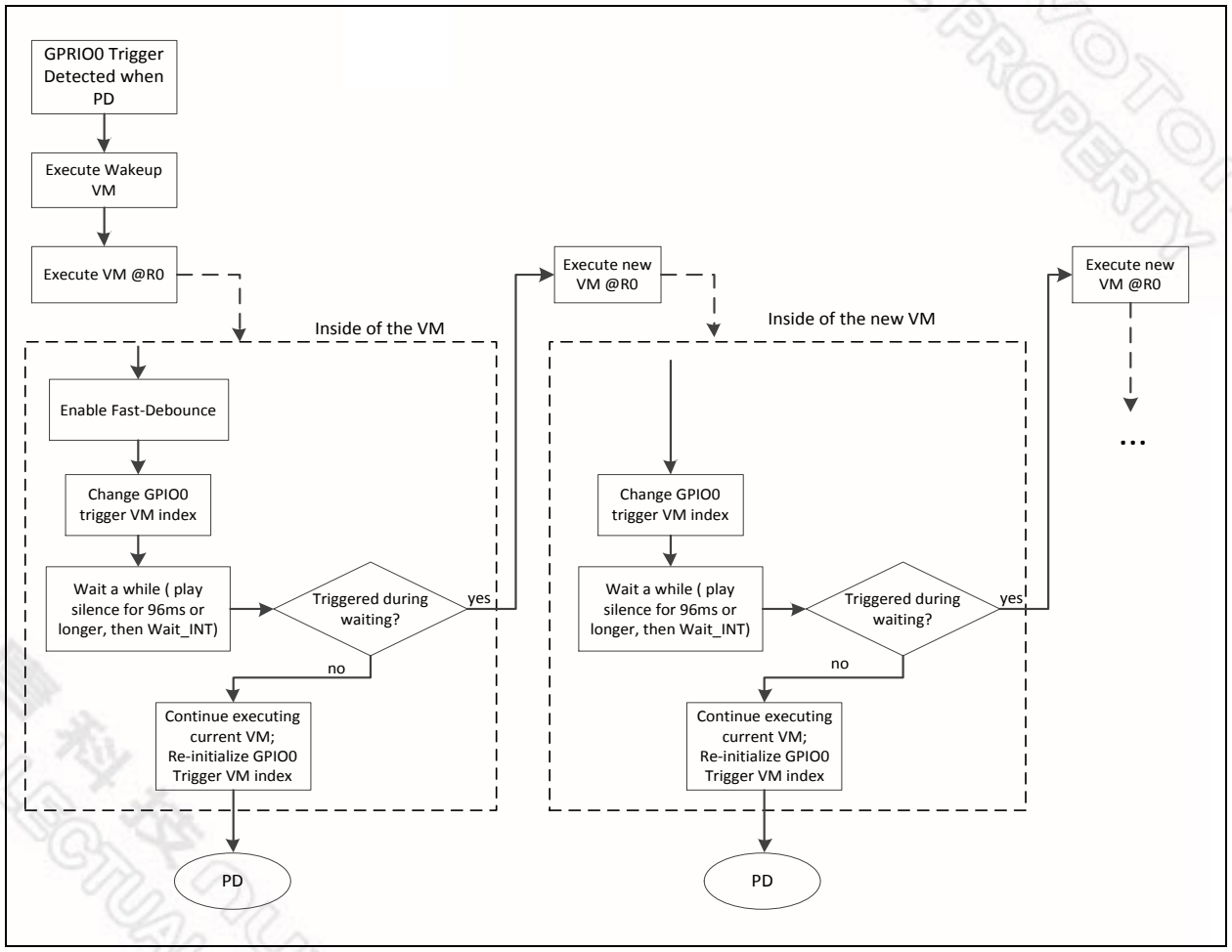

Figure 2-1 One-Pin Control Voice Macro Execution Flow

- Sample Voice Macro

  There are four kinds Voice Macro to be implemented. The following is the explanation of the role for each kind of VM.

  POI VM – Configure the falling edge trigger on GPIO0 pin; initialize the trigger VM index.

  Wakeup VM – Enable fast de-bounce.

  Stop VM – Increase trigger index, give waiting window, re-initialize the trigger index.

  Task VM – Increase the trigger index, gives waiting window, re-initialize the trigger index before play a VP.

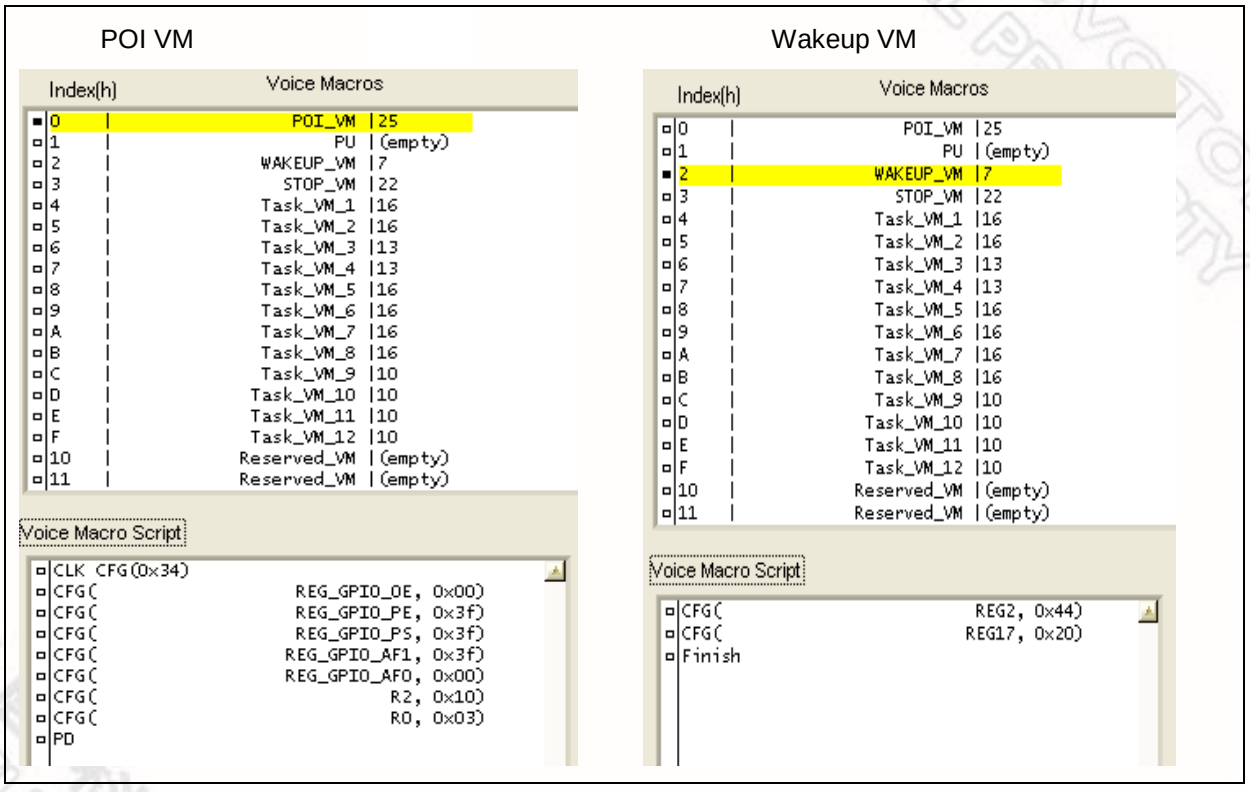

Figure 2-2 Sample POI VM and Wakeup VM

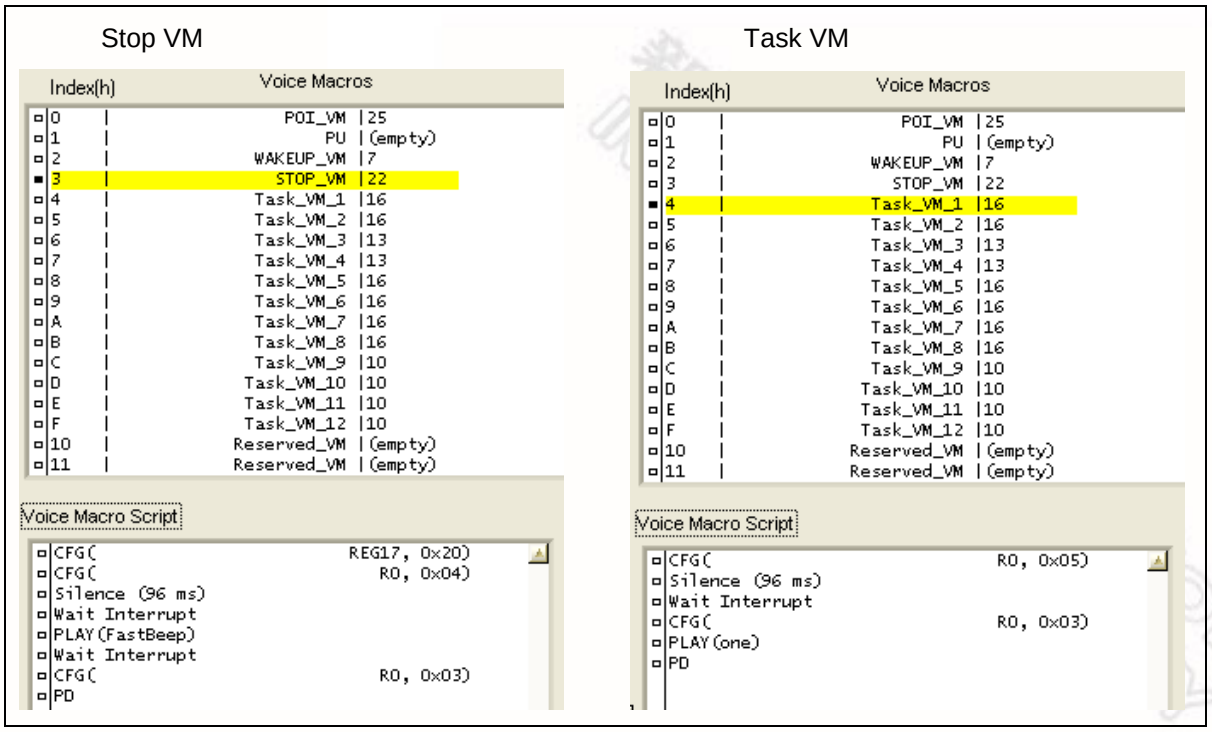

Figure 2-3 Sample Stop VM and Task VM

2.2 GPIO Trigger Timing Control

This section defines the triggering timing.

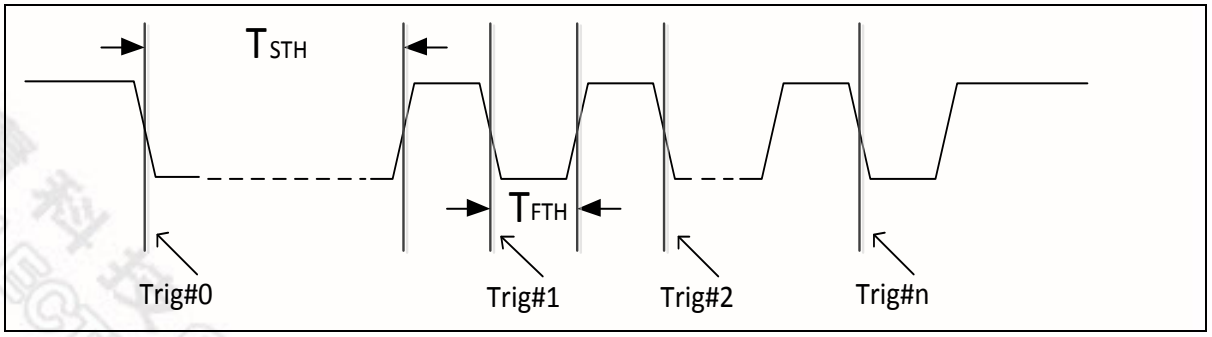

Figure 2-4 One-Pin Control Voice Macro Execution Flow

Parameter	Symbol	MIN	TYP	MAX	Unit	Remark
Slow De-Bounce Holding Time	T_{STH}	~20	30	-	ms	Normal De-Bounce
Fast De-Bounce Holding Time	T_{FTH}	50		-	µs	

Table 2-2 Triggering Timing specification

The first trigger referred as Trig#0 should always use slow de-bounce time which is about 30ms, because by default from PD state the normal (slow) de-bounce is in effect. The triggers following that should be all fast de-bounce triggers, and the de-bounce holding time can be >50 µs.

To initialize the internal triggering index, the MCU can issue one trigger using long slow de-bounce holding time. Thus it's guaranteed that a STOP VM can be successfully invoked to run. Also, if a VM is executing and when its VP is playing, issuing only one GPIO trigger can stop that VM and re-initialize the trigger index.

As a summary, the one-pin trigger VP/VM playback control is done by this:

- Issue one trigger to stop, i.e. to initialize the trigger index.
- Issue (n+1) triggers will play the VM with index n. The first trigger has slow de-bounce time; while all the following n triggers use fast de-bounce time.

2.3 Event processing

Figure 2-5 further describes the underlying idea of one-pin trigger processing. The on board MCU sends a sequence of pulses to trigger the event. The duty cycle for each pulse can be 50%.

Trig#0 uses the slow de-bounce time, so the MCU needs to hold IO low for 30ms. Once the VM machine is running, within the VM waiting window, the on board MCU needs to issue the following trigger again if a jumping is desired; otherwise a Stop/Initialization will be executed. The VM waiting window size can be very flexible, as long as user can make sure that the jumping trigger can happen in that window.

All the jumping triggers use fast de-bounce holding time, which needs to be >50 µs for good reliability. Once the jumping stopped, and the waiting window is passed, then the device will execute the VM associated with the last jumping. The VM will re-initialize the trigger index before PD the device.

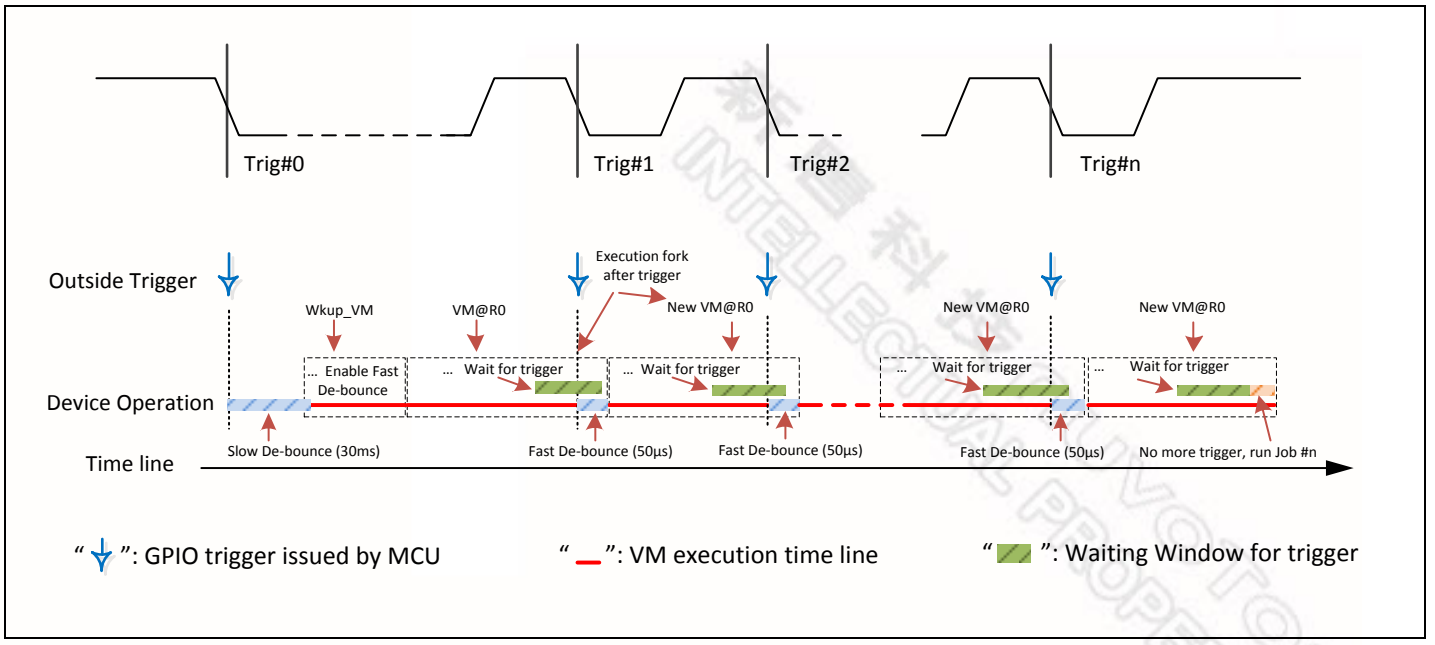

Figure 2-5 Event processing time line

3. Revision History

VERSION	DATE	REMARK
Ver_0.1	Oct 24, 2012	Initial draft
Ver_0.2	Oct 26, 2012	Description Update