Trying to use I2S example

Locked

- Print view

Search this topic…

Grumpy Mike

Posts: 1005
Joined: Sat Sep 10, 2011 7:49 pm
Location: English Lake District

Trying to use I2S example

Sun Mar 28, 2021 2:56 pm

I have been trying to copy the I2S version of the sine_wave program. I used Pico Tools to set up the system and copied the sin_wave.c file into my new directory and changed the name of it to my new directory. I have used this technique before successfully for outputting sound on an D/A connected to the SPI bus.

but when running make at the line:-

> Code:
>
> ```
> #include "pico/audio_i2s.h"
> ```

I get a no such file or directory error.
Changing the links or moving the files clears that error but generates another one as the audio_i2s.h #includes other stuff. I can go down the rabbit hole

until I am stuck.

I have come to the conclusion there must be something I am missing in the (to my mind), totally unexplained make files that are littered about the place.

Could some one help me please on finding out what to change to get this example working.

shabtronic

Posts: 82
Joined: Mon Feb 08, 2021 11:13 pm

Re: Trying to use I2S example

Sun Mar 28, 2021 3:04 pm

have you changed your CMakeLists.txt to include pico_audio_i2s ?

something like:

target_link_libraries(pio_st7789_lcd PRIVATE pico_stdlib pico_audio_i2s tinyusb_host tinyusb_board pico_multicore hardware_dma hardware_flash hardware_pll hardware_vreg hardware_clocks hardware_adc hardware_pio hardware_spi hardware_interp)

Grumpy Mike

Posts: 1005
Joined: Sat Sep 10, 2011 7:49 pm
Location: English Lake District

Re: Trying to use I2S example

Sun Mar 28, 2021 3:07 pm

Thanks, yes I have done that. But it makes no difference.

shabtronic

Posts: 82
Joined: Mon Feb 08, 2021 11:13 pm

Re: Trying to use I2S example

Sun Mar 28, 2021 3:09 pm

Ahh ok - I did have problems relocating the examples to a new directories.

So I just kept the examples in their original directories and just hacked em up. And that's because I don't really understand the cmake system properly at the moment.

Grumpy Mike

Posts: 1005
Joined: Sat Sep 10, 2011 7:49 pm
Location: English Lake District

Re: Trying to use I2S example

Sun Mar 28, 2021 3:14 pm

To be specific my error message is:-

> Code:
>
> ```
> pi@raspberrypi400:~/Tone_I2S/build $ make
> [  3%] Built target bs2_default
> [  5%] Built target bs2_default_bin
> [  6%] Built target bs2_default_padded_checksummed_asm
> [  8%] Performing build step for 'ELF2UF2Build'
> [100%] Built target elf2uf2
> [ 10%] No install step for 'ELF2UF2Build'
> [ 11%] Completed 'ELF2UF2Build'
> [ 20%] Built target ELF2UF2Build
> [ 22%] Building C object CMakeFiles/Tone_I2S.dir/Tone_I2S.c.obj
> /home/pi/Tone_I2S/Tone_I2S.c:18:10: fatal error: pico/audio_i2s.h: No such file or directory
>  #include "pico/audio_i2s.h"
>           ^~~~~~~~~~~~~~~~~~
> compilation terminated.
> make[2]: *** [CMakeFiles/Tone_I2S.dir/build.make:63: CMakeFiles/Tone_I2S.dir/Tone_I2S.c.obj] Error 1
> make[1]: *** [CMakeFiles/Makefile2:74: CMakeFiles/Tone_I2S.dir/all] Error 2
> make: *** [Makefile:84: all] Error 2
> ```

and the last bit of my cmake file says:-

> Code:
>
> ```
> # Add the standard library to the build
> target_link_libraries(Tone_I2S
>     pico_stdlib
>     pico_audio_i2s
>     )
> ```

> And that's because I don't really understand the cmake system properly at the moment.

Thanks, glad I am not the only one.

shabtronic
Posts: 82
Joined: Mon Feb 08, 2021 11:13 pm

Re: Trying to use I2S example

Sun Mar 28, 2021 3:18 pm

I put the I2S example code into the st7789 pio example, because I was experimenting with lcds and audio e.t.c..

here's my windows 10 command line output from that project :

> Code:
>
> ```
> d:\TheVault\Pico\Examples\build\pio\st7789_lcd>nmake
>
> Microsoft (R) Program Maintenance Utility Version 14.27.29112.0
> Copyright (C) Microsoft Corporation.  All rights reserved.
>
> [  0%] Built target bs2_default
> [  0%] Built target bs2_default_padded_checksummed_asm
> [  0%] Performing build step for 'PioasmBuild'
>
> Microsoft (R) Program Maintenance Utility Version 14.27.29112.0
> Copyright (C) Microsoft Corporation.  All rights reserved.
>
> [100%] Built target pioasm
> [  0%] No install step for 'PioasmBuild'
> [  0%] Completed 'PioasmBuild'
> [  0%] Built target PioasmBuild
> [  0%] Built target pico_audio_i2s_audio_i2s_pio_h
> [  0%] Performing build step for 'ELF2UF2Build'
>
> Microsoft (R) Program Maintenance Utility Version 14.27.29112.0
> Copyright (C) Microsoft Corporation.  All rights reserved.
>
> [100%] Built target elf2uf2
> [  0%] No install step for 'ELF2UF2Build'
> [  0%] Completed 'ELF2UF2Build'
> [  0%] Built target ELF2UF2Build
> [  0%] Built target pio_st7789_lcd_st7789_lcd_pio_h
> Scanning dependencies of target pio_st7789_lcd
> [  0%] Building C object pio/st7789_lcd/CMakeFiles/pio_st7789_lcd.dir/st7789_lcd.c.obj
> [  0%] Linking CXX executable pio_st7789_lcd.elf
> [100%] Built target pio_st7789_lcd
>
> d:\TheVault\Pico\Examples\build\pio\st7789_lcd>
> ```

Grumpy Mike

Posts: 1005
Joined: Sat Sep 10, 2011 7:49 pm
Location: English Lake District

Re: Trying to use I2S example

Sun Mar 28, 2021 3:22 pm

> here's my windows 10 command line output from that project

Thanks again. However I am not using windows but a Raspberry Pi 400.

shabtronic

Posts: 82
Joined: Mon Feb 08, 2021 11:13 pm

Re: Trying to use I2S example

Sun Mar 28, 2021 3:27 pm

yeah - sure I know that - just showing what the output should be.

I suspect the pico-extras-master hasn't installed properly - I can't remember how I installed that tho, but that's what it looks like.

I vaguely remember having to faff about with some files to get the extras stuff to work:

I have these environ vars:

PICO_EXTRAS_PATH=D:\TheVault\Pico\pico-extras-master
PICO_SDK_PATH=D:\TheVault\pico\pico-sdk-master

and in

Pico\Examples\CMakeLists.txt

I have

> Code:
>
> ```
> cmake_minimum_required(VERSION 3.12)
>
> # Pull in SDK (must be before project)
> include(pico_sdk_import.cmake)
>
> include(pico_extras_import.cmake)
> ```

```
project(pico_examples C CXX ASM)
set(CMAKE_C_STANDARD 11)
set(CMAKE_CXX_STANDARD 17)

set(PICO_EXAMPLES_PATH ${PROJECT_SOURCE_DIR})

# Initialize the SDK
pico_sdk_init()

include(example_auto_set_url.cmake)
# Add blink example
add_subdirectory(blink)

# Add hello world example
add_subdirectory(hello_world)

# Hardware-specific examples in subdirectories:
add_subdirectory(adc)
add_subdirectory(clocks)
add_subdirectory(cmake)
add_subdirectory(divider)
add_subdirectory(dma)
add_subdirectory(flash)
add_subdirectory(gpio)
add_subdirectory(i2c)
add_subdirectory(interp)
add_subdirectory(multicore)
add_subdirectory(picoboard)
add_subdirectory(pio)
add_subdirectory(pwm)
add_subdirectory(reset)
add_subdirectory(rtc)
add_subdirectory(spi)
add_subdirectory(system)
add_subdirectory(timer)
add_subdirectory(uart)
add_subdirectory(usb)
add_subdirectory(watchdog)
```

Grumpy Mike

Posts: 1005
Joined: Sat Sep 10, 2011 7:49 pm
Location: English Lake District

Re: Trying to use I2S example

Sun Mar 28, 2021 5:53 pm

Thanks, but sadly no joy.

We could do with a good document describing what the cmake files do and how they should be used.

shabtronic

Posts: 82
Joined: Mon Feb 08, 2021 11:13 pm

Re: Trying to use I2S example

Sun Mar 28, 2021 5:57 pm

I think for me - the environ var and include(pico_extras_import.cmake) were the things missing from the extras install - that got me working.

Yeah I agree - I wish I had more time to fully understand the cmake system - it's probably super simple, but I just don't have the time.

We'll I hope you figure out your issuse - good luck!

~~kilograham~~

Raspberry Pi Engineer & Forum Moderator

Posts: 1368
Joined: Fri Apr 12, 2019 11:00 am
Location: austin tx

Re: Trying to use I2S example

Sun Mar 28, 2021 7:28 pm

Did you do what it says at the bottom of the README here https://github.com/raspberrypi/pico-ext ... /README.md (https://github.com/raspberrypi/pico-extras/blob/master/README.md)

you should rename **pico_playground** to the name of your project (that example was just copied from pico_playground it seems)

There is information of the cmake build here https://datasheets.raspberrypi.org/pico ... -c-sdk.pdf (https://datasheets.raspberrypi.org/pico/raspberry-pi-pico-c-sdk.pdf) (chapter 2.1). It is short as there isn't really much to know to just use it, otherwise there is plenty of cmake documentation on the internet if you want to dig in further.

One hint, if your setup is screwed up in some way, then definitely delete and recreate your build directory before rebuilding.

Grumpy Mike

Posts: 1005
Joined: Sat Sep 10, 2011 7:49 pm
Location: English Lake District

Re: Trying to use I2S example

Tue Mar 30, 2021 6:45 am

Thank you for your reply. However, I am not sure about some things.

> you should rename pico_playground to the name of your project

I do think that is not a long term solution, eventually I would like to share my work as open source example. As I see it tying things to the way you have installed Pico on your specific machine cuts down the number of people who could make use of it.

As it is even a simple LED blink project is 4.4 MB (5.6 MB on disk) and contains 667 files, for a 20 line C program. Is it me or is this totally insane?

So what I have done is to make an "include" directory alongside my main .c file. In this I put the file that was being complained about the audio_i2c.h file and changed my main code to include that file from there. Of course this fails because the audio_i2c.h calls up audio.h, so I change where audio_i2c.h is looking for this file and put it in my include directory. This then causes further errors because I can't find the buffer.h file so I edit the audio.h file and put the buffer.h file in my include directory.

The result of this is it now compiles whoopee!

However it now falls down at the link phase boo!
with the following:-

> Code:
>
> ```
> [ 98%] Building C object CMakeFiles/Tone_I2S.dir/home/pi/pico/pico-sdk/src/rp2_common/pico_stdio_uart/stdio_uart.c.obj
> [100%] Linking CXX executable Tone_I2S.elf
> /usr/lib/gcc/arm-none-eabi/7.3.1/../../../arm-none-eabi/bin/ld: cannot find -lpico_audio_i2s
> collect2: error: ld returned 1 exit status
> make[2]: *** [CMakeFiles/Tone_I2S.dir/build.make:672: Tone_I2S.elf] Error 1
> make[1]: *** [CMakeFiles/Makefile2:74: CMakeFiles/Tone_I2S.dir/all] Error 2
> make: *** [Makefile:84: all] Error 2
> ```

Is this fixable given what I have done?

kilograham

Raspberry Pi Engineer & Forum Moderator

ENGINEER

Posts: 1368
Joined: Fri Apr 12, 2019 11:00 am
Location: austin tx

Re: Trying to use I2S example

Tue Mar 30, 2021 9:33 pm

> *Grumpy Mike (./memberlist.php?mode=viewprofile&u=1772&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1844153&sid=5128ba85f44b07c66111f3a7028abce5#p1844153)*
> *Tue Mar 30, 2021 6:45 am*
>
> Thank you for your reply. However, I am not sure about some things.
>
> > you should rename pico_playground to the name of your project
>
> I do think that is not a long term solution, eventually I would like to share my work as open source example. As I see it tying things to the way you have installed Pico on your specific machine cuts down the number of people who could make use of it.

i meant in the **project(pico_playground)** which is in the README.md example I pointed you at... i.e. copy that example, but give it your project name (not copy all of pico_playground and rename it).

kilograham

Raspberry Pi Engineer & Forum Moderator

ENGINEER

Posts: 1368
Joined: Fri Apr 12, 2019 11:00 am
Location: austin tx

Re: Trying to use I2S example

Tue Mar 30, 2021 9:36 pm

> *Grumpy Mike (./memberlist.php?mode=viewprofile&u=1772&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1844153&sid=5128ba85f44b07c66111f3a7028abce5#p1844153)*
> *Tue Mar 30, 2021 6:45 am*
>
> As it is even a simple LED blink project is 4.4 MB (5.6 MB on disk) and contains 667 files, for a 20 line C program. Is it me or is this totally insane?

What do you mean by a "project"... your source files should be a few K only!??

> So what I have done is to make an "include" directory alongside my main .c file. In this I put the file that was being complained about the audio_i2c.h file and changed my main code to include that file from there. Of course this fails because the audio_i2c.h calls up audio.h, so I change where audio_i2c.h is looking for this file and put it in my include directory. This then causes further errors because I can't find the buffer.h file so I edit the audio.h file and put the buffer.h file in my include directory.
>
> The result of this is it now compiles whoopee!
>
> However it now falls down at the link phase boo!
> with the following:-
>
> > Code:
> >
> > ```
> > [ 98%] Building C object CMakeFiles/Tone_I2S.dir/home/pi/pico/pico-sdk/src/rp2_common/pico_stdio_uart/stdio_uart.c.obj
> > [100%] Linking CXX executable Tone_I2S.elf
> > /usr/lib/gcc/arm-none-eabi/7.3.1/../../../arm-none-eabi/bin/ld: cannot find -lpico_audio_i2s
> > collect2: error: ld returned 1 exit status
> > make[2]: *** [CMakeFiles/Tone_I2S.dir/build.make:672: Tone_I2S.elf] Error 1
> > make[1]: *** [CMakeFiles/Makefile2:74: CMakeFiles/Tone_I2S.dir/all] Error 2
> > make: *** [Makefile:84: all] Error 2
> > ```
>
> Is this fixable given what I have done?

start over with the basic example from the README.md I suggested above

~~hippy~~

Posts: 12904
Joined: Fri Sep 09, 2011 10:34 pm
Location: UK

Re: Trying to use I2S example

Wed Mar 31, 2021 4:43 pm

> *kilograham (./memberlist.php?mode=viewprofile&u=294330&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1844481&sid=5128ba85f44b07c66111f3a7028abce5#p1844481)*
> *Tue Mar 30, 2021 9:36 pm*
>
> _

> > *Grumpy Mike (./memberlist.php?mode=viewprofile&u=1772&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1844153&sid=5128ba85f44b07c66111f3a7028abce5#p1844153)*
> > *Tue Mar 30, 2021 6:45 am*
> >
> > As it is even a simple LED blink project is 4.4 MB (5.6 MB on disk) and contains 667 files, for a 20 line C program. Is it me or is this totally insane?
>
> What do you mean by a "project"... your source files should be a few K only!??

I would guess he means how running 'cmake' then 'make' expands the disk usage under the project root directory from a few K to 4 or 5 MB -

> Code:
>
> ```
> pi@Pi3B:~/mypico/myblink $ ls
> CMakeLists.txt  myblink.c
> pi@Pi3B:~/mypico/myblink $ mkdir build
> pi@Pi3B:~/mypico/myblink $ cd build
> pi@Pi3B:~/mypico/myblink/build $ du -h -d 1 ..
> 4.0K    ../build
> 16K     ..
> pi@Pi3B:~/mypico/myblink/build $ cmake ..
> ...
> pi@Pi3B:~/mypico/myblink/build $ du -h -d 1 ..
> 2.6M    ../build
> 2.6M    ..
> pi@Pi3B:~/mypico/myblink/build $ make
> ...
> pi@Pi3B:~/mypico/myblink/build $ du -h -d 1 ..
> 4.2M    ../build
> 4.2M    ..
> ```

It's perhaps not a lot in the grand scheme of things but is a bit of a trial for those of us used to 'gcc -o file file.c'; do we tidy up as we go to maintain minimalist ideals and suffer every build recreating those MB with all the extra wear on SD card that entails, or live with all those build files.

I have added a 'cmake clean' option which suits minimalism ...

> Code:
>
> ```
> pi@Pi3B:~/mypico/myblink $ ls
> CMakeLists.txt  myblink.c
> pi@Pi3B:~/mypico/myblink $ du -h -d 1 .
> 12K     .
> pi@Pi3B:~/mypico/myblink $ cmake .
> ...
> pi@Pi3B:~/mypico/myblink $ make
> ...
> pi@Pi3B:~/mypico/myblink $ ls
> CMakeCache.txt       CMakeLists.txt  Makefile     myblink.dis      myblink.hex
> CMakeFiles           elf2uf2         myblink.bin  myblink.elf      myblink.uf2
> cmake_install.cmake  generated       myblink.c    myblink.elf.map  pico-sdk
> pi@Pi3B:~/mypico/myblink $ du -h -d 1 .
> 1.8M    ./pico-sdk
> 20K     ./generated
> 652K    ./elf2uf2
> 1.1M    ./CMakeFiles
> 4.0M    .
> pi@Pi3B:~/mypico/myblink $ cmake clean
> pi@Pi3B:~/mypico/myblink $ ls
> CMakeLists.txt  myblink.c  myblink.uf2
> ```

```
pi@Pi3B:~/mypico/myblink $ du -h -d 1 .
32K     .
```

I still don't understand why we need to build 'elf2uf2' every project, takes over half a MB for that. It does seem insane that this is not a system app like any other. Same too for 'pioasm'. I was half expecting it to build the 'gcc' compilers and 'make' from source for good measure.

I was thinking some way to build using a RAM Disk /tmp might go someway to a solution of achieving a balance but haven't figured out how to do that.

My own attempt to copy and build 'sine_wave' out of tree, the same way as worked with 'blink' as above; gives me "unknown type name 'audio_format_t'" so guess I'm missing some include.

~~hippy~~

Posts: 12904
Joined: Fri Sep 09, 2011 10:34 pm
Location: UK

Re: Trying to use I2S example

Wed Mar 31, 2021 5:58 pm

> *Grumpy Mike (./memberlist.php?mode=viewprofile&u=1772&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1843415&sid=5128ba85f44b07c66111f3a7028abce5#p1843415)*
> *Sun Mar 28, 2021 3:14 pm*
>
> > And that's because I don't really understand the cmake system properly at the moment.
>
> Thanks, glad I am not the only one.

That makes at least three of us.

This worked for me so I hope it might help. Straight copy of 'sine_wave.c' from the playground, my own 'CMakeLists.txt', and note the hard-wired 'include' ...

~/mypico/mysine_wave/CMakeLists.txt

> Code:
>
> ```
> cmake_minimum_required(VERSION 3.12)
> set(CMAKE_C_STANDARD 11)
> set(CMAKE_CXX_STANDARD 17)
> set(PICO_SDK_PATH /home/pi/pico/pico-sdk)
> include(~/pico/pico-sdk/external/pico_sdk_import.cmake)
> include(~/pico/pico-extras/external/pico_extras_import.cmake)
> project(mysine_wave C CXX)
> pico_sdk_init()
> add_executable(mysine_wave sine_wave.c)
> target_compile_definitions(mysine_wave PRIVATE USE_AUDIO_I2S=1 PICO_AUDIO_I2S_MONO_INPUT=1 )
> target_link_libraries(mysine_wave pico_stdlib pico_audio_i2s)
> pico_add_extra_outputs(mysine_wave)
> pico_enable_stdio_usb(mysine_wave 1)
> ```

> Code:
>
> ```
> pi@Pi3B:~/mypico/mysine_wave $ ls
> CMakeLists.txt  sine_wave.c
> ```

```
pi@Pi3B:~/mypico/mysine_wave $ cmake .
...
pi@Pi3B:~/mypico/mysine_wave $ make
...
[100%] Linking CXX executable mysine_wave.elf
[100%] Built target mysine_wave
pi@Pi3B:~/mypico/mysine_wave $ ls
CMakeCache.txt       generated        mysine_wave.elf.map  pioasm
CMakeFiles           Makefile         mysine_wave.hex      sine_wave.c
cmake_install.cmake  mysine_wave.bin  mysine_wave.uf2
CMakeLists.txt       mysine_wave.dis  pico_extras
elf2uf2              mysine_wave.elf  pico-sdk
```

Also works in a 'build' directory with 'cmake ..'

kilograham

Raspberry Pi Engineer & Forum Moderator

ENGINEER

Posts: 1368
Joined: Fri Apr 12, 2019 11:00 am
Location: austin tx

Re: Trying to use I2S example

Wed Mar 31, 2021 7:46 pm

> *hippy (./memberlist.php?mode=viewprofile&u=1729&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1844845&sid=5128ba85f44b07c66111f3a7028abce5#p1844845)*
> *Wed Mar 31, 2021 5:58 pm*
>
> This worked for me so I hope it might help. Straight copy of 'sine_wave.c' from the playground, my own 'CMakeLists.txt', and note the hard-wired 'include' ...
>
> That looks good - I guess user preference whether you copy the pico_xxx_import.cmake local to your project or not (it only serves to find the XXX based on the PICO_XXX_PATH env var/build var etc.) - note from the linked README.md in pico-extras "You can add Pico Extras to your project similarly to the SDK (copying external/pico_extras_import.cmake into your project) having set the PICO_EXTRAS_PATH variable in your environment or via cmake variable."

kilograham

Raspberry Pi Engineer & Forum Moderator

ENGINEER

Posts: 1368
Joined: Fri Apr 12, 2019 11:00 am
Location: austin tx

Re: Trying to use I2S example

Wed Mar 31, 2021 7:55 pm

> *hippy (./memberlist.php?mode=viewprofile&u=1729&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1844810&sid=5128ba85f44b07c66111f3a7028abce5#p1844810)*
> *Wed Mar 31, 2021 4:43 pm*
>
> _

> *kilograham (./memberlist.php?mode=viewprofile&u=294330&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1844481&sid=5128ba85f44b07c66111f3a7028abce5#p1844481)*
> *Tue Mar 30, 2021 9:36 pm*
>
> > *Grumpy Mike (./memberlist.php?mode=viewprofile&u=1772&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1844153&sid=5128ba85f44b07c66111f3a7028abce5#p1844153)*
> > *Tue Mar 30, 2021 6:45 am*
> >
> > As it is even a simple LED blink project is 4.4 MB (5.6 MB on disk) and contains 667 files, for a 20 line C program. Is it me or is this totally insane?
>
> What do you mean by a "project"... your source files should be a few K only!??

I would guess he means how running 'cmake' then 'make' expands the disk usage under the project root directory from a few K to 4 or 5 MB - ...
It's perhaps not a lot in the grand scheme of things but is a bit of a trial for those of us used to 'gcc -o file file.c'; do we tidy up as we go to maintain minimalist ideals and suffer every build recreating those MB with all the extra wear on SD card that entails, or live with all those build files.

I am all for not wasting space, but I find it very hard to believe that a few meg of build files is an issue for anyone. Many installations of things are 1G+ nowadays.

elf2uf2 does not need to be rebuilt every time unless you keep deleting it. we decided to build it from within the SDK then we don't have to worry about having to get it correctly installed on Linux, macOS, Windows. The more steps involved, the more people manage to shoot themselves in the foot

You can certainly build to a temporary disk if you like.

We build all the SDK library code from source too, because we can make smaller, faster code that way for your deliverable. You are free to write programs with a binary editor if you prefer

~~hippy~~

Posts: 12904
Joined: Fri Sep 09, 2011 10:34 pm
Location: UK

Re: Trying to use I2S example

Fri Apr 02, 2021 12:10 pm

> *kilograham (./memberlist.php?mode=viewprofile&u=294330&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1844913&sid=5128ba85f44b07c66111f3a7028abce5#p1844913)*
> *Wed Mar 31, 2021 7:55 pm*
>
> You can certainly build to a temporary disk if you like.

I have got that working, can now type 'cmake ram' which works for me, but not sure if this is the best way to do it ...

> Code:
>
> ```
> HOMEDIR=$(pwd)
> if [ "$HOMEDIR" != "" ]
> then
>   BASEDIR=$(basename -- $FILENAME $HOMEDIR)
>   if [ "$BASEDIR" != "" ]
>   then
>     if [ -f CMakeLists.txt ]
>     then
> ```

```
        mkdir -p /tmp/cmake/$BASEDIR
        cp -r .  /tmp/cmake/$BASEDIR
        cd       /tmp/cmake/$BASEDIR
        cmake .
        make
        cp *.uf2 $HOMEDIR/
      fi
    fi
  fi
```

Is there any documentation on how one should do it ?

kilograham

Raspberry Pi Engineer & Forum Moderator

ENGINEER

Posts: 1368
Joined: Fri Apr 12, 2019 11:00 am
Location: austin tx

Re: Trying to use I2S example

Fri Apr 02, 2021 5:36 pm

you don't need to copy anything, you just create your build directory on the temporary drive.

the

> Code:
>
> ```
> mkdir build
> cd build
> cmake ..
> ```

Is set in stone; build can be anywhere, and replace the .. with the relative (or absolute) path to the source.
cmake options are -S and -B to specify source and build directories; check the cmake documentation

Grumpy Mike

Posts: 1005
Joined: Sat Sep 10, 2011 7:49 pm
Location: English Lake District

Re: Trying to use I2S example

Sun Apr 04, 2021 4:28 am

Thanks hippy, your reply - 31 Mar 2021 17:58 worked very well for me. For the sake of anyone wanting to know the answer this is it. To make it work on the Pimoroni pico audio board you have to change the sin_wave.c (or what ever you call it) to read:-

> Code:
>
> ```
>     struct audio_i2s_config config = {
>             // for Pimoroni pico audio board
> ```

```
            .data_pin = 9,
            .clock_pin_base = 10,
            .dma_channel = 0,
            .pio_sm = 0,
    };
```

Or leave that alone and edit the file /home/pi/pico/pico-extras/src/rp2_common/pico_audio_i2s/include/pico to change the #defines concerning the pins once and for all:-

Code:

```
#ifndef PICO_AUDIO_I2S_DATA_PIN
//#warning PICO_AUDIO_I2S_DATA_PIN should be defined when using AUDIO_I2S
//#define PICO_AUDIO_I2S_DATA_PIN 27
#define PICO_AUDIO_I2S_DATA_PIN 9
#endif

#ifndef PICO_AUDIO_I2S_CLOCK_PIN_BASE
//#warning PICO_AUDIO_I2S_CLOCK_PIN_BASE should be defined when using AUDIO_I2S
//#define PICO_AUDIO_I2S_CLOCK_PIN_BASE 25
#define PICO_AUDIO_I2S_CLOCK_PIN_BASE 10
#endif
```

I think the former is better.

****

As to the wider discussion, while being mainly a hardware guy that programs, in a couple of jobs I had, I also had to manage software engineers. I appreciate the desire for saving time with only compiling what needs to be compiled when dealing with large projects. The sort of projects that you make overnight and has multiple engineers working on it at the same time. You don't change project very often, you spend months and perhaps years with the same project. I think the Pico C programming offerings of the Raspberry Foundation are built with that mindset by people with that sort of background.

This is the exact opposite of what you need for a small micro controller, especially one that is meant to encourage learning. The programs are short, they have to be in order to fit into the controller, they are changed and compiled often and only one person is working on it. They are modified and changed and renamed. It seems that simply renaming a project is one hell of a hassle.

To that end the Arduino guys have it right. Yes I know the argument is that they have had years of doing it, and we should all wait until they bring out their software, but it entirely possible that they will rig it so that you will not be able to program an RP2040 board with their IDE. They were bitter about the Teensy board and the way that their software "rescued" that project to the detriment of their income. (I know this from personal discussions with the founders of the Arduino project) and you still can't program a Teensy without Paul producing an overlay / patch for each new IDE version. I know nothing about the contract they signed with the Raspberry Pi foundation, but it might have a bearing on what sort of software support they give to the processor in the form of our board.

In one folder on the Arduino on my Mac, I have 1,372 programs / projects and that occupies just 12MB, contrast that with a single blink and LED program that takes up 4.4MB and has 667 files associated with it. I would suggest that this matters in the overall scheme of things, especially over the long term.

~~kilograham~~

Raspberry Pi Engineer & Forum Moderator

ENGINEER

Posts: 1368
Joined: Fri Apr 12, 2019 11:00 am
Location: austin tx

Re: Trying to use I2S example

Sun Apr 04, 2021 1:01 pm

> *Grumpy Mike (./memberlist.php?mode=viewprofile&u=1772&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1846248&sid=5128ba85f44b07c66111f3a7028abce5#p1846248)*
> *Sun Apr 04, 2021 4:28 am*
>
> Or leave that alone and edit the file /home/pi/pico/pico-extras/src/rp2_common/pico_audio_i2s/include/pico to change the #defines concerning the pins once and for all:-
>
> > Code:
> >
> > ```
> > #ifndef PICO_AUDIO_I2S_DATA_PIN
> > //#warning PICO_AUDIO_I2S_DATA_PIN should be defined when using AUDIO_I2S
> > //#define PICO_AUDIO_I2S_DATA_PIN 27
> > #define PICO_AUDIO_I2S_DATA_PIN 9
> > #endif
> >
> > #ifndef PICO_AUDIO_I2S_CLOCK_PIN_BASE
> > //#warning PICO_AUDIO_I2S_CLOCK_PIN_BASE should be defined when using AUDIO_I2S
> > //#define PICO_AUDIO_I2S_CLOCK_PIN_BASE 25
> > #define PICO_AUDIO_I2S_CLOCK_PIN_BASE 10
> > #endif
> > ```

You never need to change the headers in the pico SDK. You can override the variables such as PICO_AUDIO_I2S_CLOCK_PIN_BASE for your application from a board configuration header, or from the CMakeLists,.txt for your app.

~~hippy~~

Posts: 12904
Joined: Fri Sep 09, 2011 10:34 pm
Location: UK

Re: Trying to use I2S example

Sun Apr 04, 2021 1:28 pm

> *Grumpy Mike (./memberlist.php?mode=viewprofile&u=1772&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1846248&sid=5128ba85f44b07c66111f3a7028abce5#p1846248)*
> *Sun Apr 04, 2021 4:28 am*
>
> I appreciate the desire for saving time with only compiling what needs to be compiled when dealing with large projects ... This is the exact opposite of what you need for a small micro controller ... The programs are short, they have to be in order to fit into the controller, they are changed and compiled often and only one person is working on it.

I appreciate what you are saying and for genuinely small microcontrollers with a simple compiler or assembler, and little in the way of a build system, I would also agree that 'build it all' isn't usually a problem. That's the way I usually work.

But the RP2040 is far from small, at least not in the way people are encouraged to use the C SDK rather than treat it as bare metal.

I don't have a problem with the C SDK approach, building it all when needed in an optimal manner, only building what is needed as and when. Nor much problem with the 'huge disk usage' as that can easily be cleaned out if one doesn't want to keep it, don't mind 'build it all' next time. And it can all be built on RAM disk or /tmp so it doesn't ever appear in one's own project tree.

Hacking about with MicroPython, which often requires a complete rebuild, the 3 minutes it takes to do that on my Pi 3B has quickly become a PITA. Not helped that it can be 2 or more minutes before reporting I've mistyped 'count' as 'cont' or something worse. I'd much prefer it if it didn't require everything to be built. So I can't support 'build it all' as the best way.

If one does want 'build it all' every time it's easy enough to do 'rm -f -r build' and have that.

> *Grumpy Mike (./memberlist.php?mode=viewprofile&u=1772&sid=5128ba85f44b07c66111f3a7028abce5) wrote: ↑ (./viewtopic.php?p=1846248&sid=5128ba85f44b07c66111f3a7028abce5#p1846248)*
> *Sun Apr 04, 2021 4:28 am*
>
> It seems that simply renaming a project is one hell of a hassle.

It can be quite a hassle initially extracting stuff to one's own project tree from 'pico-extras', 'pico-examples', and 'pico-playground', can be a PITA when something requires something one hasn't done before - as here. I walked the exact same path as you, ran into the very same obstacles and frustrations, spent quite some time trying to make what I ended up with work.

But I have found that once one has figured it out for one it's mostly the same for the next. Most projects I have built in my own tree, Raspberry Pi provided and from others, has simply been putting them in my own tree, copying CMakeLists.txt from a working project and changing the name of the old to the new in that. Yes, this one needed stuff adding which I wasn't familiar with but I now know how to do any other using the same, can use this CMakeList.txt as its template.

Most of my projects consist only of 'CMakeLists.txt" and "project.c" and stay that way with only "project.uf2" added after building and I am happy with that.

It took some effort to figure out how to do that so I could put it in a shell script so it's as simple as 'cmake make' but it has repaid that effort.

> Code:
>
> ```
> pi@Pi3B:~/mypico/mysine_wave $ ls
> CMakeLists.txt  sine_wave.c
> pi@Pi3B:~/mypico/mysine_wave $ cmake make
> ... 100 seconds of waiting ...
> pi@Pi3B:~/mypico/mysine_wave $ ls
> CMakeLists.txt  sine_wave.c  sine_wave.uf2
> ```

- Other languages
- Deutsch
- Español
- Français
- Italiano
- Nederlands
- 日本語
- Polski
- Português
- Русский
- Türkçe
- User groups and events
- The MagPi
- Using the Raspberry Pi
- Beginners
- Troubleshooting
- Advanced users
- Assistive technology and accessibility
- Education
- Picademy
- Teaching and learning resources
- Staffroom, classroom and projects
- Astro Pi
- Mathematica
- High Altitude Balloon
- Weather station
- Programming
- C/C++
- Java
- Python
- Scratch
- Other programming languages
- Windows 10 for IoT
- Wolfram Language
- Bare metal, Assembly language
- Graphics programming
- OpenGLES
- OpenVG
- OpenMAX
- General programming discussion
- Projects
- Networking and servers
- Automation, sensing and robotics
- Graphics, sound and multimedia
- Other projects
- Gaming
- Media centres
- AIY Projects
- Hardware and peripherals
- Camera board
- Compute Module
- Official Display
- HATs and other add-ons
- Device Tree
- Interfacing (DSI, CSI, I2C, etc.)
- Raspberry Pi 400
- Raspberry Pi Pico
- General
- SDK
- MicroPython

- Other RP2040 boards
- Operating system distributions
- Raspberry Pi OS
- Raspberry Pi Desktop for PC and Mac
- Other
- Android
- Debian
- FreeBSD
- Gentoo
- Linux Kernel
- NetBSD
- openSUSE
- Plan 9
- Puppy
- Arch
- Pidora / Fedora
- RISCOS
- Ubuntu
- Ye Olde Pi Shoppe
- For sale
- Wanted
- Off topic
- Off topic discussion

- Board index
- All times are UTC
- Delete cookies

Follow us

Sign up to newsletter

Twitter

Mastodon

YouTube

Instagram

LinkedIn

Facebook

About Raspberry Pi

News

Contact us

Trademark

About us

Jobs

Accessibility

Cookies

Licensing

Terms and conditions of sale

Privacy

Security

Verify our bank details

For home

Raspberry Pi for home

Tutorials

For industry

Raspberry Pi for industry

Thin clients

Powered by Raspberry Pi

Design partners

Success stories

Hardware

Computers and microcontrollers

Cameras and displays

Add-on boards

Power supplies and cables

Cases

Peripherals

Software

Raspberry Pi OS

Raspberry Pi Imager

Raspberry Pi Desktop

Documentation

All categories

Product information portal

Datasheets

Forums

All forums

Raspberry Pi Store

Cambridge Store

Raspberry Pi Press

About Raspberry Pi Press

The MagPi

HackSpace

Wireframe

Custom PC