

list7.txt

入力:W25Q32のアクセス部分を、次のプログラムを元に修正してください。

```
/**
 * Copyright (c) 2020 Raspberry Pi (Trading) Ltd.
 *
 * SPDX-License-Identifier: BSD-3-Clause
 */

// Example of reading/writing an external serial flash using the PL022 SPI
interface

#include <stdio.h>
#include "pico/stdlib.h"
#include "pico/binary_info.h"
#include "hardware/spi.h"

#define FLASH_PAGE_SIZE      256
#define FLASH_SECTOR_SIZE   4096

#define FLASH_CMD_PAGE_PROGRAM 0x02
#define FLASH_CMD_READ         0x03
#define FLASH_CMD_STATUS       0x05
#define FLASH_CMD_WRITE_EN     0x06
#define FLASH_CMD_SECTOR_ERASE 0x20

#define FLASH_STATUS_BUSY_MASK 0x01

static inline void cs_select(uint cs_pin) {
    asm volatile("nop \n nop \n nop"); // FIXME
    gpio_put(cs_pin, 0);
    asm volatile("nop \n nop \n nop"); // FIXME
}

static inline void cs_deselect(uint cs_pin) {
    asm volatile("nop \n nop \n nop"); // FIXME
    gpio_put(cs_pin, 1);
    asm volatile("nop \n nop \n nop"); // FIXME
}

void __not_in_flash_func(flash_read)(spi_inst_t *spi, uint cs_pin, uint32_t
addr, uint8_t *buf, size_t len) {
    cs_select(cs_pin);
    uint8_t cmdbuf[4] = {
        FLASH_CMD_READ,
        addr >> 16,
        addr >> 8,
        addr
    };
    spi_write_blocking(spi, cmdbuf, 4);
    spi_read_blocking(spi, 0, buf, len);
    cs_deselect(cs_pin);
}
```

list7.txt

```
void __not_in_flash_func(flash_write_enable)(spi_inst_t *spi, uint cs_pin) {
    cs_select(cs_pin);
    uint8_t cmd = FLASH_CMD_WRITE_EN;
    spi_write_blocking(spi, &cmd, 1);
    cs_deselect(cs_pin);
}

void __not_in_flash_func(flash_wait_done)(spi_inst_t *spi, uint cs_pin) {
    uint8_t status;
    do {
        cs_select(cs_pin);
        uint8_t buf[2] = {FLASH_CMD_STATUS, 0};
        spi_write_read_blocking(spi, buf, buf, 2);
        cs_deselect(cs_pin);
        status = buf[1];
    } while (status & FLASH_STATUS_BUSY_MASK);
}

void __not_in_flash_func(flash_sector_erase)(spi_inst_t *spi, uint cs_pin,
uint32_t addr) {
    uint8_t cmdbuf[4] = {
        FLASH_CMD_SECTOR_ERASE,
        addr >> 16,
        addr >> 8,
        addr
    };
    flash_write_enable(spi, cs_pin);
    cs_select(cs_pin);
    spi_write_blocking(spi, cmdbuf, 4);
    cs_deselect(cs_pin);
    flash_wait_done(spi, cs_pin);
}

void __not_in_flash_func(flash_page_program)(spi_inst_t *spi, uint cs_pin,
uint32_t addr, uint8_t data[]) {
    uint8_t cmdbuf[4] = {
        FLASH_CMD_PAGE_PROGRAM,
        addr >> 16,
        addr >> 8,
        addr
    };
    flash_write_enable(spi, cs_pin);
    cs_select(cs_pin);
    spi_write_blocking(spi, cmdbuf, 4);
    spi_write_blocking(spi, data, FLASH_PAGE_SIZE);
    cs_deselect(cs_pin);
    flash_wait_done(spi, cs_pin);
}

void printbuf(uint8_t buf[FLASH_PAGE_SIZE]) {
    for (int i = 0; i < FLASH_PAGE_SIZE; ++i) {
        if (i % 16 == 15)
```

```

                                list7.txt
        printf("%02x\n", buf[i]);
    else
        printf("%02x ", buf[i]);
    }
}

int main() {
    // Enable UART so we can print status output
    stdio_init_all();
#if !defined(spi_default) || !defined(PICO_DEFAULT_SPI_SCK_PIN) ||
!defined(PICO_DEFAULT_SPI_TX_PIN) || !defined(PICO_DEFAULT_SPI_RX_PIN) ||
!defined(PICO_DEFAULT_SPI_CSN_PIN)
#warning spi/spi_flash example requires a board with SPI pins
    puts("Default SPI pins were not defined");
#else

    printf("SPI flash example\n");

    // Enable SPI 0 at 1 MHz and connect to GPIOs
    spi_init(spi_default, 1000 * 1000);
    gpio_set_function(PICO_DEFAULT_SPI_RX_PIN, GPIO_FUNC_SPI);
    gpio_set_function(PICO_DEFAULT_SPI_SCK_PIN, GPIO_FUNC_SPI);
    gpio_set_function(PICO_DEFAULT_SPI_TX_PIN, GPIO_FUNC_SPI);
    // Make the SPI pins available to picotool
    bi_decl(bi_3pins_with_func(PICO_DEFAULT_SPI_RX_PIN, PICO_DEFAULT_SPI_TX_PIN,
PICO_DEFAULT_SPI_SCK_PIN, GPIO_FUNC_SPI));

    // Chip select is active-low, so we'll initialise it to a driven-high state
    gpio_init(PICO_DEFAULT_SPI_CSN_PIN);
    gpio_put(PICO_DEFAULT_SPI_CSN_PIN, 1);
    gpio_set_dir(PICO_DEFAULT_SPI_CSN_PIN, GPIO_OUT);
    // Make the CS pin available to picotool
    bi_decl(bi_1pin_with_name(PICO_DEFAULT_SPI_CSN_PIN, "SPI CS"));

    printf("SPI initialised, let's goooooo\n");

    uint8_t page_buf[FLASH_PAGE_SIZE];

    const uint32_t target_addr = 0;

    flash_sector_erase(spi_default, PICO_DEFAULT_SPI_CSN_PIN, target_addr);
    flash_read(spi_default, PICO_DEFAULT_SPI_CSN_PIN, target_addr, page_buf,
FLASH_PAGE_SIZE);

    printf("After erase:\n");
    printbuf(page_buf);

    for (int i = 0; i < FLASH_PAGE_SIZE; ++i)
        page_buf[i] = i;
    flash_page_program(spi_default, PICO_DEFAULT_SPI_CSN_PIN, target_addr,
page_buf);
    flash_read(spi_default, PICO_DEFAULT_SPI_CSN_PIN, target_addr, page_buf,

```

list7.txt

```
FLASH_PAGE_SIZE);

    printf("After program:\n");
    printbuf(page_buf);

    flash_sector_erase(spi_default, PICO_DEFAULT_SPI_CSN_PIN, target_addr);
    flash_read(spi_default, PICO_DEFAULT_SPI_CSN_PIN, target_addr, page_buf,
FLASH_PAGE_SIZE);

    printf("Erase again:\n");
    printbuf(page_buf);

    return 0;
#endif
}
```

GPT: 了解しました。あなたの要望に基づいて、フラッシュメモリW25Q32から2バイトのバイナリデータを読み出してPWMに変換するプログラムを作成します。まず、提供されたフラッシュメモリアクセス用のコードに合わせて読み取り機能を統合します。次に、PWM設定とデータの変換を行います。

以下は、統合されたコードの例です:

```
#include <stdio.h>
#include "pico/stdlib.h"
#include "pico/binary_info.h"
#include "hardware/spi.h"
#include "hardware/pwm.h"

// Flash configuration
#define FLASH_PAGE_SIZE      256
#define FLASH_CMD_READ      0x03
#define FLASH_STATUS_BUSY_MASK 0x01

// SPI configuration
#define SPI_PORT spi_default
#define CS_PIN PICO_DEFAULT_SPI_CSN_PIN

// PWM configuration
#define PWM_PIN 6
#define PWM_FREQ_HZ 16000

// Functions related to SPI flash access
void cs_select(uint cs_pin);
void cs_deselect(uint cs_pin);
void flash_read(spi_inst_t *spi, uint cs_pin, uint32_t addr, uint8_t *buf,
size_t len);

// Functions related to PWM
void setup_pwm();
void set_pwm_duty_cycle(uint16_t sample);

int main() {
    // SPI and UART initialization
    stdio_init_all();
```

list7.txt

```
spi_init(SPI_PORT, 1000 * 1000);
gpio_set_function(PICO_DEFAULT_SPI_RX_PIN, GPIO_FUNC_SPI);
gpio_set_function(PICO_DEFAULT_SPI_SCK_PIN, GPIO_FUNC_SPI);
gpio_set_function(PICO_DEFAULT_SPI_TX_PIN, GPIO_FUNC_SPI);
gpio_init(CS_PIN);
gpio_put(CS_PIN, 1);
gpio_set_dir(CS_PIN, GPIO_OUT);

// PWM initialization
setup_pwm();

uint32_t address = 0x000000;
uint8_t buf[2];

while (1) {
    flash_read(SPI_PORT, CS_PIN, address, buf, 2);
    uint16_t sample = ((uint16_t)buf[0] << 8) | buf[1];
    set_pwm_duty_cycle(sample);
    sleep_us(1000000 / PWM_FREQ_HZ);
    address += 2;
}

return 0;
}

// Functions related to SPI flash access
void cs_select(uint cs_pin) {
    asm volatile("nop \n nop \n nop");
    gpio_put(cs_pin, 0);
    asm volatile("nop \n nop \n nop");
}

void cs_deselect(uint cs_pin) {
    asm volatile("nop \n nop \n nop");
    gpio_put(cs_pin, 1);
    asm volatile("nop \n nop \n nop");
}

void flash_read(spi_inst_t *spi, uint cs_pin, uint32_t addr, uint8_t *buf,
size_t len) {
    cs_select(cs_pin);
    uint8_t cmdbuf[4] = {
        FLASH_CMD_READ,
        addr >> 16,
        addr >> 8,
        addr
    };
    spi_write_blocking(spi, cmdbuf, 4);
    spi_read_blocking(spi, 0, buf, len);
    cs_deselect(cs_pin);
}
```

list7.txt

```
// Functions related to PWM
```

```
void setup_pwm() {  
    gpio_set_function(PWM_PIN, GPIO_FUNC_PWM);  
    uint slice_num = pwm_gpio_to_slice_num(PWM_PIN);  
    pwm_set_wrap(slice_num, 0xffff);  
    pwm_set_clkdiv(slice_num, clock_get_hz(clk_sys) / (PWM_FREQ_HZ *  
0x10000ull));  
}
```

```
void set_pwm_duty_cycle(uint16_t sample) {  
    uint slice_num = pwm_gpio_to_slice_num(PWM_PIN);  
    pwm_set_chan_level(slice_num, PWM_CHAN_A, sample);  
}
```

このコードは、W25Q32から音声データを読み取り、PWM出力として再生するための基本的な実装を提供していません。コードの正確性を確保するために、適切なハードウェアと環境でテストしてください。