Bookworm vs Bullseye

Bookworm vs Bullseye

- RPi5
- cat /etc/os-release
- PRETTY_NAME="Debian GNU/Linux 12 (bookworm)"
- NAME="Debian GNU/Linux"
- VERSION_ID="12"
- VERSION="12 (bookworm)"
- VERSION_CODENAME=bookworm
- ID=debian
- HOME_URL="https://www.debian.org/"
- SUPPORT_URL="https://www.debian.org/support"
- BUG_REPORT_URL="https://bugs.debian.org/"
-

- RPi4
- cat /etc/os-release
- PRETTY_NAME="Debian GNU/Linux 11 (bullseye)"
- NAME="Debian GNU/Linux"
- VERSION_ID="11"
- VERSION="11 (bullseye)"
- VERSION_CODENAME=bullseye
- ID=debian
- HOME_URL="https://www.debian.org/"
- SUPPORT_URL="https://www.debian.org/support"
- BUG_REPORT_URL="https://bugs.debian.org/"
-

Bookworm vs Bullseye

- ii binutils-arm-none-eabi 2.35.2-2+14+b2 arm64 GNU assembler, linker and binary utilities for ARM Cortex-R/M processors
- ii gcc-arm-none-eabi 15:8-2019-q3-1+b1 arm64 GCC cross compiler for ARM Cortex-R/M processors
- ii libnewlib-arm-none-eabi 3.3.0-1 all C library and math library compiled for bare metal using Cortex A/R/M
- ii libstdc++-arm-none-eabi-newlib 15:8-2019-q3-1+13 all GNU Standard C++ Library v3 for ARM Cortex-R/M processors (newlib)
-

- ii binutils-arm-none-eabi 2.40-2+18+b1 arm64 GNU assembler, linker and binary utilities for ARM Cortex-R/M processors
- ii gcc-arm-none-eabi 15:12.2.rel1-1 arm64 GCC cross compiler for ARM Cortex-R/M processors
- ii libnewlib-arm-none-eabi 3.3.0-1.3 all C library and math library compiled for bare metal using Cortex A/R/M
- ii libstdc++-arm-none-eabi-dev 15:12.2.rel1-1+23 all GNU Standard C++ Library v3 for ARM Cortex-R/M processors (headers)
- ii libstdc++-arm-none-eabi-newlib 15:12.2.rel1-1+23 all GNU Standard C++ Library v3 for ARM Cortex-R/M processors (newlib)
-

picow_freertos_iperf.c

Uses Networking functions

Uses Circular buffer functions

Bullseye

head_tail.h

```
char tmp[80];
char * ptrhead;
char * ptrtail;
char * ptrendofbuf;
char * ptrtopofbuf;
```

tcp_debug.h

```
char client_message[BUF_SIZE];
```

head_tail.c

Circular buffer functions

tcp_debug.c

Networking functions

Bookworm

picow_freertos_iperf.c

```
char tmp[80];
char * ptrhead;
char * ptrtail;
char * ptrendofbuf;
char * ptrtopofbuf;

char client_message[BUF_SIZE]
```

Uses Networking functions

Uses Circular buffer functions

head_tail.h

head_tail.c

Circular buffer functions

tcp_debug.h

```
extern char client_message[BUF_SIZE];
```

tcp_debug.c

Networking functions

23 pico_w/wifi/freertos/iperf/head_tail.h

```
@@ -1,22 +1,17 @@
 #ifndef HEAD_TAIL_H
 #define HEAD_TAIL_H
-#include <stdio.h>
-#include "pico/stdlib.h"
-#include "pico/cyw43_arch.h"
-
-#include "lwip/pbuf.h"
-#include "lwip/tcp.h"
-char tmp[80];
+
+/*
 char * head;
 char * tail;
 char * endofbuf;
 char * topofbuf;
+*/

-char * bump_head(char * head, char * endofbuf,char * topofbuf);
-char * bump_tail(char * tail,char * endofbuf,char * topofbuf);
-char * dec_head(char * head,char * endofbuf,char * topofbuf);
-char * dec_tail(char * tail,char * endofbuf,char * topofbuf);
-char * head_tail_helper(char * head, char * tail,char * endofbuf,char * topofbuf, char * inpstr);
-
-#endif
+char * bump_head(char * ptrhead, char * ptrendofbuf,char * ptrtopofbuf);
+char * bump_tail(char * ptrtail,char * ptrendofbuf,char * ptrtopofbuf);
+char * dec_head(char * ptrhead,char * ptrendofbuf,char * ptrtopofbuf);
+char * dec_tail(char * ptrtail,char * ptrendofbuf,char * ptrtopofbuf);
+char * head_tail_helper(char * ptrhead, char * ptrtail,char * ptrendofbuf,char * ptrtopofbuf, char * inpstr);
+
+#endif
```

pico_w-mqtt / pico_w / wifi / freertos / iperf / **tcp_debug.h**

develone 0x20001584 0x20001554 0x20001753 0x20001554

Code | Blame | 38 lines (33 loc) · 1.08 KB | Code 55% faster with GitHub Copilot

```
1   #ifndef TCP_DEBUG_H
2   #define TCP_DEBUG_H
3   #include "pico/stdlib.h"
4   #include "pico/cyw43_arch.h"
5
6   #include "lwip/pbuf.h"
7   #include "lwip/tcp.h"
8
9   #define TCP_PORT 4001
10  #define DEBUG_printf printf
11  #define BUF_SIZE 512
12  #define TEST_ITERATIONS 1
13  #define POLL_TIME_S 5
14  extern char client_message[BUF_SIZE];
15
16  typedef struct TCP_SERVER_T_ {
17      struct tcp_pcb *server_pcb;
18      struct tcp_pcb *client_pcb;
19      bool complete;
20      uint8_t buffer_sent[BUF_SIZE];
21      uint8_t buffer_recv[BUF_SIZE];
22      int sent_len;
23      int recv_len;
24      int run_count;
25  } TCP_SERVER_T;
26
27  TCP_SERVER_T* tcp_server_init(void);
28  err_t tcp_server_close(void *arg);
29  err_t tcp_server_result(void *arg, int status);
30  err_t tcp_server_sent(void *arg, struct tcp_pcb *tpcb, u16_t len);
31  err_t tcp_server_send_data(void *arg, struct tcp_pcb *tpcb);
32  err_t tcp_server_recv(void *arg, struct tcp_pcb *tpcb, struct pbuf *p, err_t err);
33  err_t tcp_server_poll(void *arg, struct tcp_pcb *tpcb);
34  void tcp_server_err(void *arg, err_t err);
35  err_t tcp_server_accept(void *arg, struct tcp_pcb *client_pcb, err_t err);
36  bool tcp_server_open(void *arg);
37
38  #endif
```

```
61  typedef struct NTP_T_ {
98  u8_t reset_remote=0;
99  int val = 0;
100 int loop;
101 // This array converts a number 0-9 to a bit pattern to send to the GPIOs
102 int bits[10] = {
103         0x3f,  // 0
104         0x3e,  // 1
105         0x3d,  // 2
106         0x3c,  // 3
107         0x3b,  // 4
108         0x3a,  // 5
109         0x39,  // 6
110         0x38,  // 7
111         0x37,  // 8
112         0x36   // 9
113 };
114 // This array converts a number 0-9 to a bit pattern to send to the GPIOs
115 int32_t mask;
116
117 u8_t lp;
118 u8_t alarm_hour;
119 u8_t alarm_min;
120 u8_t alarm_sec;
121 u8_t remote_index;
122 u8_t cmd;
123 char houralarm[3];
124 char minalarm[3];
125 char secalarm[3];
126 char tmp[80];
127 char * ptrhead;
128 char * ptrtail;
129 char * ptrendofbuf;
130 char * ptrtopofbuf;
131 char client_message[BUF_SIZE];
132 mqtt_request_cb_t pub_mqtt_request_cb_t;
133
134 u16_t mqtt_port = 1883;
135
136 char PUB_PAYLOAD_T[]="                                    ";
137 char PUB_PAYLOAD_SCR_T[]="                                    ";
138 char PUB_EXTRA_ARG_T[] = "test";
139 u16_t payload_t_size;
140
141 /* References for this implementation:
142  * raspberry-pi-pico-c-sdk.pdf, Section '4.1.1. hardware_adc'
143  * pico-examples/adc/adc_console/adc_console.c */
144 float read_onboard_temperature(char unit) {
145
146     /* 12-bit conversion, assume max value == ADC_VREF == 3.3 V */
147     float conversionFactor = 3.3f / (1 << 12);
148
149     float adc = (float)adc_read() * conversionFactor;
150     float tempC = 27.0f - (adc - 0.706f) / 0.001721f;
151
152     if (unit == 'C') {
153         return tempC;
```