Listing 1. Pełny kod głównej funkcji programu – *app\_main()*

void app\_main**(**void**)**

**{**

/\* init default NVS partition \*/

ESP\_ERROR\_CHECK**(**nvs\_flash\_init**());**

/\* init LwIP stack \*/

tcpip\_adapter\_init**();**

/\* stop DHCP server/client \*/

ESP\_ERROR\_CHECK**(**tcpip\_adapter\_dhcps\_stop**(**TCPIP\_ADAPTER\_IF\_AP**));**

ESP\_ERROR\_CHECK**(**tcpip\_adapter\_dhcpc\_stop**(**TCPIP\_ADAPTER\_IF\_STA**));**

/\* event initialization \*/

ESP\_ERROR\_CHECK**(**esp\_event\_loop\_create\_default**());**

/\* Wi-Fi initialization \*/

wifi\_init\_config\_t config **=** WIFI\_INIT\_CONFIG\_DEFAULT**();**

ESP\_ERROR\_CHECK**(**esp\_wifi\_init**(&**config**));**

/\* register IP events handler \*/

ESP\_ERROR\_CHECK**(**esp\_event\_handler\_register**(**IP\_EVENT**,** IP\_EVENT\_STA\_GOT\_IP**,** **&**ip\_event\_handler**,** **NULL));**

/\* start Wi-Fi \*/

ESP\_ERROR\_CHECK**(**esp\_wifi\_set\_storage**(**WIFI\_STORAGE\_FLASH**));**

ESP\_ERROR\_CHECK**(**esp\_wifi\_start**());**

/\* mesh initialization \*/

ESP\_ERROR\_CHECK**(**esp\_mesh\_init**());**

/\* register mesh events handler \*/

ESP\_ERROR\_CHECK**(**esp\_event\_handler\_register**(**MESH\_EVENT**,** ESP\_EVENT\_ANY\_ID**,** **&**mesh\_event\_handler**,** **NULL));**

/\* initialize mesh\_cfg\_t with default values \*/

mesh\_cfg\_t cfg **=** MESH\_INIT\_CONFIG\_DEFAULT**();**

/\* mesh ID \*/

memcpy**((**uint8\_t **\*)** **&**cfg**.**mesh\_id**,** MESH\_ID**,** 6**);**

/\* router \*/

cfg**.**channel **=** MESH\_CHANNEL**;**

cfg**.**router**.**ssid\_len **=** strlen**(**MESH\_ROUTER\_SSID**);**

memcpy**((**uint8\_t **\*)** **&**cfg**.**router**.**ssid**,** MESH\_ROUTER\_SSID**,** cfg**.**router**.**ssid\_len**);**

memcpy**((**uint8\_t **\*)** **&**cfg**.**router**.**password**,** MESH\_ROUTER\_PASSWD**,** strlen**(**MESH\_ROUTER\_PASSWD**));**

/\* mesh softAP \*/

ESP\_ERROR\_CHECK**(**esp\_mesh\_set\_ap\_authmode**(**MESH\_AP\_AUTHMODE**));**

cfg**.**mesh\_ap**.**max\_connection **=** MESH\_AP\_CONNECTIONS**;**

memcpy**((**uint8\_t **\*)** **&**cfg**.**mesh\_ap**.**password**,** MESH\_AP\_PASSWD**,** strlen**(**MESH\_AP\_PASSWD**));**

/\* mesh settings \*/

ESP\_ERROR\_CHECK**(**esp\_mesh\_set\_max\_layer**(**MESH\_MAX\_LAYER**));**

ESP\_ERROR\_CHECK**(**esp\_mesh\_set\_vote\_percentage**(**1**));**

ESP\_ERROR\_CHECK**(**esp\_mesh\_set\_ap\_assoc\_expire**(**10**));**

/\* set mesh config \*/

ESP\_ERROR\_CHECK**(**esp\_mesh\_set\_config**(&**cfg**));**

/\* mesh start \*/

ESP\_ERROR\_CHECK**(**esp\_mesh\_start**());**

ESP\_LOGI**(**TAG**,** "Mesh starts successfully: %s\n"**,** esp\_mesh\_is\_root\_fixed**()** **?** "root fixed" **:** "root not fixed"**);**

**}**

Listing 2. Pełny kod funkcji obsługi zdarzeń protokołu ESP-MESH

static int mesh\_layer **=** **-**1**;**

static mesh\_addr\_t mesh\_parent\_addr**;**

void mesh\_event\_handler**(**void **\***arg**,** esp\_event\_base\_t event\_base**,**

int32\_t event\_id**,** void **\***event\_data**)**

**{**

mesh\_addr\_t id **=** **{**0**,};**

static uint8\_t last\_layer **=** 0**;**

**switch** **(**event\_id**)** **{**

**case** MESH\_EVENT\_STARTED**:** **{**

esp\_mesh\_get\_id**(&**id**);**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_MESH\_STARTED>ID:"MACSTR""**,** MAC2STR**(**id**.**addr**));**

mesh\_layer **=** esp\_mesh\_get\_layer**();**

**}**

**break;**

**case** MESH\_EVENT\_STOPPED**:** **{**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_STOPPED>"**);**

mesh\_layer **=** esp\_mesh\_get\_layer**();**

**}**

**break;**

**case** MESH\_EVENT\_CHILD\_CONNECTED**:** **{**

mesh\_event\_child\_connected\_t **\***child\_connected **=** **(**mesh\_event\_child\_connected\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_CHILD\_CONNECTED>aid:%d, "MACSTR""**,**

child\_connected**->**aid**,**

MAC2STR**(**child\_connected**->**mac**));**

**}**

**break;**

**case** MESH\_EVENT\_CHILD\_DISCONNECTED**:** **{**

mesh\_event\_child\_disconnected\_t **\***child\_disconnected **=** **(**mesh\_event\_child\_disconnected\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_CHILD\_DISCONNECTED>aid:%d, "MACSTR""**,**

child\_disconnected**->**aid**,**

MAC2STR**(**child\_disconnected**->**mac**));**

**}**

**break;**

**case** MESH\_EVENT\_ROUTING\_TABLE\_ADD**:** **{**

mesh\_event\_routing\_table\_change\_t **\***routing\_table **=** **(**mesh\_event\_routing\_table\_change\_t **\*)**event\_data**;**

ESP\_LOGW**(**TAG**,** "<MESH\_EVENT\_ROUTING\_TABLE\_ADD>add %d, new:%d"**,**

routing\_table**->**rt\_size\_change**,**

routing\_table**->**rt\_size\_new**);**

**}**

**break;**

**case** MESH\_EVENT\_ROUTING\_TABLE\_REMOVE**:** **{**

mesh\_event\_routing\_table\_change\_t **\***routing\_table **=** **(**mesh\_event\_routing\_table\_change\_t **\*)**event\_data**;**

ESP\_LOGW**(**TAG**,** "<MESH\_EVENT\_ROUTING\_TABLE\_REMOVE>remove %d, new:%d"**,**

routing\_table**->**rt\_size\_change**,**

routing\_table**->**rt\_size\_new**);**

**}**

**break;**

**case** MESH\_EVENT\_NO\_PARENT\_FOUND**:** **{**

mesh\_event\_no\_parent\_found\_t **\***no\_parent **=** **(**mesh\_event\_no\_parent\_found\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_NO\_PARENT\_FOUND>scan times:%d"**,**

no\_parent**->**scan\_times**);**

**}**

**break;**

**case** MESH\_EVENT\_PARENT\_CONNECTED**:** **{**

mesh\_event\_connected\_t **\***connected **=** **(**mesh\_event\_connected\_t **\*)**event\_data**;**

esp\_mesh\_get\_id**(&**id**);**

mesh\_layer **=** connected**->**self\_layer**;**

memcpy**(&**mesh\_parent\_addr**.**addr**,** connected**->**connected**.**bssid**,** 6**);**

ESP\_LOGI**(**TAG**,**

"<MESH\_EVENT\_PARENT\_CONNECTED>layer:%d-->%d, parent:"MACSTR"%s, ID:"MACSTR""**,**

last\_layer**,** mesh\_layer**,** MAC2STR**(**mesh\_parent\_addr**.**addr**),**

esp\_mesh\_is\_root**()** **?** "<ROOT>" **:**

**(**mesh\_layer **==** 2**)** **?** "<layer2>" **:** ""**,** MAC2STR**(**id**.**addr**));**

last\_layer **=** mesh\_layer**;**

**if** **(**esp\_mesh\_is\_root**())** **{**

tcpip\_adapter\_dhcpc\_start**(**TCPIP\_ADAPTER\_IF\_STA**);**

**}**

**}**

**break;**

**case** MESH\_EVENT\_PARENT\_DISCONNECTED**:** **{**

mesh\_event\_disconnected\_t **\***disconnected **=** **(**mesh\_event\_disconnected\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,**

"<MESH\_EVENT\_PARENT\_DISCONNECTED>reason:%d"**,**

disconnected**->**reason**);**

mesh\_layer **=** esp\_mesh\_get\_layer**();**

**}**

**break;**

**case** MESH\_EVENT\_LAYER\_CHANGE**:** **{**

mesh\_event\_layer\_change\_t **\***layer\_change **=** **(**mesh\_event\_layer\_change\_t **\*)**event\_data**;**

mesh\_layer **=** layer\_change**->**new\_layer**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_LAYER\_CHANGE>layer:%d-->%d%s"**,**

last\_layer**,** mesh\_layer**,**

esp\_mesh\_is\_root**()** **?** "<ROOT>" **:**

**(**mesh\_layer **==** 2**)** **?** "<layer2>" **:** ""**);**

last\_layer **=** mesh\_layer**;**

**}**

**break;**

**case** MESH\_EVENT\_ROOT\_ADDRESS**:** **{**

mesh\_event\_root\_address\_t **\***root\_addr **=** **(**mesh\_event\_root\_address\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_ROOT\_ADDRESS>root address:"MACSTR""**,**

MAC2STR**(**root\_addr**->**addr**));**

**}**

**break;**

**case** MESH\_EVENT\_VOTE\_STARTED**:** **{**

mesh\_event\_vote\_started\_t **\***vote\_started **=** **(**mesh\_event\_vote\_started\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,**

"<MESH\_EVENT\_VOTE\_STARTED>attempts:%d, reason:%d, rc\_addr:"MACSTR""**,**

vote\_started**->**attempts**,**

vote\_started**->**reason**,**

MAC2STR**(**vote\_started**->**rc\_addr**.**addr**));**

**}**

**break;**

**case** MESH\_EVENT\_VOTE\_STOPPED**:** **{**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_VOTE\_STOPPED>"**);**

**}**

**break;**

**case** MESH\_EVENT\_ROOT\_SWITCH\_REQ**:** **{**

mesh\_event\_root\_switch\_req\_t **\***switch\_req **=** **(**mesh\_event\_root\_switch\_req\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,**

"<MESH\_EVENT\_ROOT\_SWITCH\_REQ>reason:%d, rc\_addr:"MACSTR""**,**

switch\_req**->**reason**,**

MAC2STR**(** switch\_req**->**rc\_addr**.**addr**));**

**}**

**break;**

**case** MESH\_EVENT\_ROOT\_SWITCH\_ACK**:** **{**

/\* new root \*/

mesh\_layer **=** esp\_mesh\_get\_layer**();**

esp\_mesh\_get\_parent\_bssid**(&**mesh\_parent\_addr**);**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_ROOT\_SWITCH\_ACK>layer:%d, parent:"MACSTR""**,** mesh\_layer**,** MAC2STR**(**mesh\_parent\_addr**.**addr**));**

**}**

**break;**

**case** MESH\_EVENT\_TODS\_STATE**:** **{**

mesh\_event\_toDS\_state\_t **\***toDs\_state **=** **(**mesh\_event\_toDS\_state\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_TODS\_REACHABLE>state:%d"**,** **\***toDs\_state**);**

**}**

**break;**

**case** MESH\_EVENT\_ROOT\_FIXED**:** **{**

mesh\_event\_root\_fixed\_t **\***root\_fixed **=** **(**mesh\_event\_root\_fixed\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_ROOT\_FIXED>%s"**,**

root\_fixed**->**is\_fixed **?** "fixed" **:** "not fixed"**);**

**}**

**break;**

**case** MESH\_EVENT\_ROOT\_ASKED\_YIELD**:** **{**

mesh\_event\_root\_conflict\_t **\***root\_conflict **=** **(**mesh\_event\_root\_conflict\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,**

"<MESH\_EVENT\_ROOT\_ASKED\_YIELD>"MACSTR", rssi:%d, capacity:%d"**,**

MAC2STR**(**root\_conflict**->**addr**),**

root\_conflict**->**rssi**,**

root\_conflict**->**capacity**);**

**}**

**break;**

**case** MESH\_EVENT\_CHANNEL\_SWITCH**:** **{**

mesh\_event\_channel\_switch\_t **\***channel\_switch **=** **(**mesh\_event\_channel\_switch\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_CHANNEL\_SWITCH>new channel:%d"**,** channel\_switch**->**channel**);**

**}**

**break;**

**case** MESH\_EVENT\_SCAN\_DONE**:** **{**

mesh\_event\_scan\_done\_t **\***scan\_done **=** **(**mesh\_event\_scan\_done\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_SCAN\_DONE>number:%d"**,**

scan\_done**->**number**);**

**}**

**break;**

**case** MESH\_EVENT\_NETWORK\_STATE**:** **{**

mesh\_event\_network\_state\_t **\***network\_state **=** **(**mesh\_event\_network\_state\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_NETWORK\_STATE>is\_rootless:%d"**,**

network\_state**->**is\_rootless**);**

**}**

**break;**

**case** MESH\_EVENT\_STOP\_RECONNECTION**:** **{**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_STOP\_RECONNECTION>"**);**

**}**

**break;**

**case** MESH\_EVENT\_FIND\_NETWORK**:** **{**

mesh\_event\_find\_network\_t **\***find\_network **=** **(**mesh\_event\_find\_network\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_FIND\_NETWORK>new channel:%d, router BSSID:"MACSTR""**,**

find\_network**->**channel**,** MAC2STR**(**find\_network**->**router\_bssid**));**

**}**

**break;**

**case** MESH\_EVENT\_ROUTER\_SWITCH**:** **{**

mesh\_event\_router\_switch\_t **\***router\_switch **=** **(**mesh\_event\_router\_switch\_t **\*)**event\_data**;**

ESP\_LOGI**(**TAG**,** "<MESH\_EVENT\_ROUTER\_SWITCH>new router:%s, channel:%d, "MACSTR""**,**

router\_switch**->**ssid**,** router\_switch**->**channel**,** MAC2STR**(**router\_switch**->**bssid**));**

**}**

**break;**

**default:**

ESP\_LOGI**(**TAG**,** "unknown id:%d"**,** event\_id**);**

**break;**

**}**

**}**