Technologia GSM w elektronice (7)

Open AT – Zdalna aktualizacja aplikacji

Poprzednio w kursie programowania modemów AirPrime Q26 firmy Sierra Wireless po raz pierwszy w naszej aplikacji użyliśmy biblioteki internetowej WIP (Wavecom IP). W tym odcinku, przedstawiając możliwość zdalnej aktualizacji modemu, również skorzystamy z funkcji oferowanych przez tę bibliotekę.

Dodatkowe informacje:
Więcej informacji na temat produktów Sierra Wireless można znaleźć na stronach producenta: www.sierrawireless.com lub kontaktując się z firmą ACTE Sp. z o.o., która jest oficjalnym dystrybutorem opisywanych produktów oraz zapewnia pełne wsparcie techniczne.

Możliwość aktualizacji zdalnej to funkcjonalność, która pozwala na zdalne wprowadzenie zmian w oprogramowaniu urządzenia bez konieczności osobistej wizyty na miejscu instalacji. Jeśli na etapie projektowania zaimplementujemy taką możliwość, to być może dzięki niej w przyszłości zaoszczędzimy na wydatkach związanych z serwisowaniem urządzenia.

System operacyjny, pod kontrolą którego pracują moduły *AirPrime Q26* oraz *AirLink Fastrack Xtend*, jest przystosowany do zdalnej aktualizacji aplikacji użytkownika i całego firmware'u. Pozwala to nie tylko na serwis oprogramowania użytkownika, ale również umożliwia zachowanie kompatybilności modułów z siecią GSM przez długie lata dzięki wymianie firmware'u na uwzględniający zmiany w funkcjonowaniu sieci.

Do aktualizacji przeznaczone są pliki .dwl. Mogą one zawierać aplikację lub firmware. Po pobraniu pliku aktualizacji system operacyjny sprawdza jego poprawność, a następnie przystępuje do instalacji w odpowiednim obszarze pamięci. Następnie wykonywane są zerowanie i restart urządzenia.

Na **listingu 1** zamieszczono przykładową aplikację, pozwalającą na zdalne pobranie oprogramowania. Przed jej uruchomieniem należy przygotować serwer FTP, a następnie umieścić na nim plik, który zostanie pobrany przez modem. Należy zmienić parametry FTP zapisane w przykładzie programu na takie, które ma serwer używany do aktualizacji.

Aplikacja wykorzystuje bibliotekę WIP, a więc trzeba dodać ją do projektu (sposób dodania opisano w poprzednim odcinku kursu). Po uruchomieniu aplikacja wpisuje PIN do karty SIM oraz subskrybuje się do serwisu SMS. Po otrzymaniu SMS'a z nazwą pliku (np. Hello_world.dwl) następuje przygotowanie pamięci Application&Data, w której zapisywana będzie aplikacja w trakcie pobierania. Po zakończeniu procesu przygotowania pamięci (zdarzenie *ADL_AD_EVENT_RECOMPACT_DONE*) zostaje nawiązana sesja GPRS i następuje połączenie z serwerem FTP za pomocą funkcji *wip_FTPCreateOpts()*. Po nawiązaniu połączenia jako handler wywoływana jest funkcja *ftp_SessionHandler()* ze zdarzeniem *WIP_CEV_OPEN*. W niej za pomocą funkcji *wip_getFile()* rozpoczyna się pobieranie pliku.

W przykładzie zakładamy, że plik do pobrania znajduje się w katalogu głównym na serwerze, jednak gdyby tak nie było, to przed rozpoczęciem pobierania należałoby zmienić katalog umieszczenia pliku za pomocą funkcji *wip_cwd()*, a następnie poczekać na zdarzenie *WIP_CEV_DONE*.

Pobieranie pliku jest realizowane przez funkcję *ftp_DataHandler()*. Jeśli jest ona wywoływana przez system ze zdarzeniem *WIP_CEV_OPEN*, to następuje zapis odczytanych danych do celi w pamięci A&D. Wywołanie ze zdarzeniem *WIP_CEV_PEER_CLOSE* oznacza, że pobieranie pliku zostało zakończone. Finalizowana jest wtedy cela A&D, która została wcześniej zasubskrybowana z parametrem *UNDEF*, a następnie na numer, z którego została przysłana nazwa pliku, jest wysłany SMS z informacją o pomyślnym przebiegu aktualizacji. Ostatnim poleceniem jest ustawienie jednorazowego timera (T=10 s). Jego zadaniem jest instalacja nowej aplikacji. Ten sam proces nastąpi wcześniej, jeśli SMS z komunikatem zostanie wysłany poprawnie. Zastosowanie timera, to dodatkowy mechanizm, zabezpieczający przed sytuacją, w której wysyłanie SMS-a nie powiedzie się. Instalacja aplikacji jest wykonywana po wywołaniu systemowej funkcji *adl_adInstall()*.

Większość funkcji użytych w aplikacji zamieszczonej na listingu 1 została omówiona w poprzednich odcinkach kursu. Wyjątkiem są tutaj funkcje dotyczące serwisu FTP. Ich dokładniejszy opis znajduje się w dokumentacji biblioteki WIP. Jest ona dostępna po zainstalowaniu oprogramowania *M2M Studio* w menu: Help –> Help Contents –> Plug-ins Documentation –> WIP Open AT Plug-in Package.

Powyższy przykład bez problemu poradzi sobie z pobraniem nowej wersji aplikacji jak i nowej wersji firmware. W obu przypadkach są to pliki z rozszerzeniem .dwl. Jednak jeśli aktualizujemy firmware modemu, należałoby również zaktualizować aplikację na taką, która została skompilowana z właściwą dla danego firmware'u biblioteką ADL (OS). Wtedy najwygodniejszym rozwiązaniem jest połączenie plików firmware'u i pliku aplikacji użytkownika w jeden plik .dwl. Do tego celu można posłużyć się dowolnym edytorem plików binarnych. Zilustruje to poniższy przykład aktualizacji oprogramowania modułu Q2687RD.

Najpierw tworzymy w edytorze np. plik o nazwie *DWL.dwl*. Następnie na jego końcu dodajemy plik *R7.43.0_q2687RD.dwl*, a po nim – plik aplikacji. Uzyskany zbiór umieszczamy na serwerze FTP. Taki sposób przygotowania pliku spowoduje, że po pobraniu moduł Q2687RD dokona aktualizacji firmware'u i aplikacji.

Pliki z firmware można znaleźć w katalogu oprogramowania M2M Studio. Na przykład wersja R7.43 jest umieszczona w katalogu:

C:\Program Files\Sierra Wireless\Embedded Software \com.wavecom.openat.ide.spm.fw.model.7.43.0.201003261552_7.43.0.201003261552\resources

Adrian Chrzanowski
Acte Sp. z o.o.

Listing 1. Przykładowy program z zaimplementowaną możliwością aktualizacji oprogramowania

```
#include "adl_global.h"
#include "wip.h"
const u16 wm_apmCustomStackSize = 4096;
#define GPRS_PINCODE       ""
#define GPRS_APN  "erainternet"
#define GPRS_LOGIN  "erainternet"
#define GPRS_PASSWORD  "erainternet"
#define FTP_SERVER  "srv.acte.pl"
#define FTP_LOGIN  "ftp_haslo"
#define FTP_PASSWORD  "@ftp_login"

static wip_bearer_t gprs_br   = NULL;
s32 Cell_Handle;
s8 handSms;
wip_channel_t ftp_c, ftp_d;
bool zajety, aplikacja_pobrana= FALSE;
int a=0;

ascii ftp_buf[1024];
int ftp_nread;
ascii filename[32];
ascii nr_tel[20];
int udp_nread;
u32 recsize;

s8 ret;
ascii RspStr[128];
u8 Mode = ADL_SMS_MODE_TEXT;
adl_tmr_t        * TimerH;

void TimerHandler(u8 id, void *Context)
{
        adl_adInstall(Cell_Handle);
}

void ftp_Read( void)
{
while( (udp_nread = wip_read( ftp_d, ftp_buf, sizeof(ftp_buf))) > 0)
        {
                recsize += udp_nread;

                ret = adl_adWrite(Cell_Handle,udp_nread,(void*)ftp_buf);
                adl_atSendResponse(ADL_AT_RSP,".");
        }
}

void ftp_DataHandler( wip_event_t *ev, void *ctx)
{
        switch( ev->kind) {
                case WIP_CEV_OPEN:
                break;
        case WIP_CEV_READ:
                ftp_Read();
                break;
        case WIP_CEV_WRITE:
                break;
        case WIP_CEV_PEER_CLOSE:
                adl_atSendResponse ( ADL_AT_RSP, "plik pobrany - instalacja...\r\n");
                ret = adl_adFinalise(Cell_Handle);
                aplikacja_pobrana = TRUE;
                adl_smsSend(handSms,nr_tel,"aktualizacja zakonczona powodzeniem",ADL_SMS_MODE_TEXT);
                adl_tmrSubscribe(FALSE, 100, ADL_TMR_TYPE_100MS, TimerHandler);
                break;

        case WIP_CEV_ERROR:    // Close session channel
                if( ftp_c != NULL)
                {
                        wip_close( ftp_c);
                        ftp_c = NULL;
                }
                break;
        }
}

void ftp_SessionHandler( wip_event_t *ev, void *ctx)
{
        switch( ev->kind) {
        case WIP_CEV_OPEN:
                adl_atSendResponse ( ADL_AT_RSP, "Polaczenie FTP nawiazane\r\n");
                adl_atSendResponse( ADL_AT_RSP, "Pobieranie nowej wersji aplikacji\r\n");
                ftp_d = wip_getFile( ftp_c, filename, ftp_DataHandler, NULL);
                if( ftp_d == NULL)
                {   // FTP error
                        wip_close( ftp_c);
                        ftp_c = NULL;
                }
                break;
        case WIP_CEV_PEER_CLOSE:
        case WIP_CEV_ERROR:
                wip_close( ftp_c);
                ftp_c = NULL;

                adl_atSendResponse ( ADL_AT_RSP, "\n\rAktualizacja zakonczona bledem" );
                adl_smsSend(handSms,nr_tel,"aktualizacja zakonczona bledem",ADL_SMS_MODE_TEXT);
                break;
        }
}

void openFTP (void )
{
        ftp_c = wip_FTPCreateOpts( FTP_SERVER, ftp_SessionHandler, NULL, WIP_COPT_USER, FTP_LOGIN, WIP_COPT_PASSWORD,
FTP_PASSWORD, WIP_COPT_PASSIVE, 1, WIP_COPT_TYPE, 'I', WIP_COPT_PEER_PORT,  21, WIP_COPT_END );
```

Listing 1. c.d.

```
        if( ftp_c == NULL)
        {   // FTP error
                adl_atSendStdResponseExt ( ADL_AT_RSP, ADL_STR_CME_ERROR, 3 );
        }
        else
        {   // FTP started
                //adl_atSendStdResponse ( ADL_AT_RSP, ADL_STR_OK );
        }
}

void Restart_bearer()
{
        ret = wip_bearerStart( gprs_br);
        wm_sprintf( RspStr, "Restart_bearer=%d\r\n", ret);
        adl_atSendResponse( ADL_AT_RSP, RspStr);
}

static void evh_bearer( wip_bearer_t br, s8 event, void *ctx)
{
        wip_in_addr_t ip;
        ascii ipStr[16];
        switch( event)
        {
                case WIP_BEV_IP_CONNECTED:
                        wip_bearerGetOpts( br,
                                                WIP_BOPT_IP_ADDR,  &ip,
                                                WIP_BOPT_END);
                        wip_inet_ntoa( ip, ipStr, sizeof(ipStr));
                        wm_sprintf( RspStr, "+GPRS: CONNECTED IP=%s\r\n", ipStr);
                        adl_atSendResponse( ADL_AT_RSP, RspStr);
                        adl_atSendResponse( ADL_AT_RSP, „Nawiazywanie polaczenia FTP\r\n");
                openFTP();
                break;
                case WIP_BEV_IP_DISCONNECTED:
                        adl_atSendResponse( ADL_AT_RSP, "+GPRS: DISCONNECTED\r\n");
                        break;
                case WIP_BEV_STOPPED:
                        adl_atSendResponse( ADL_AT_RSP, "+GPRS: STOPPED\r\n");
                        break;
                case WIP_BEV_CONN_FAILED:
                        adl_atSendResponse( ADL_AT_RSP, "+GPRS: FAILED\r\n");
                break;
        }
        if (event != WIP_BEV_IP_CONNECTED) adl_tmrSubscribe ( FALSE, 10, ADL_TMR_TYPE_100MS, (adl_tmrHandler_t)
Restart_bearer);
}

void openGPRS(void)
{
        // otwieranie sesji GPRS
        ret = wip_bearerOpen( &gprs_br, "GPRS", evh_bearer, NULL);
        ret = wip_bearerSetOpts(gprs_br, WIP_BOPT_GPRS_APN, GPRS_APN, WIP_BOPT_LOGIN, GPRS_LOGIN, WIP_BOPT_PASSWORD,
GPRS_PASSWORD, WIP_BOPT_END);
        ret = wip_bearerStart( gprs_br);
}

void evh_AD ( adl_adEvent_e Event, u32 Progress )
{
        if(Event == ADL_AD_EVENT_RECOMPACT_DONE)
        {
                Cell_Handle = adl_adSubscribe (1,ADL_AD_SIZE_UNDEF);
                adl_atSendResponse( ADL_AT_UNS, "A&D OK, start pobierania...\r\n");
                zajety=TRUE;
                openGPRS();
        }
}

void init_AD_memory(void)
{
        ret = adl_adEventSubscribe( evh_AD); //
        wm_sprintf( RspStr, "adl_adEventSubscribe: %d", ret);
        TRACE((2,RspStr));
        Cell_Handle= adl_adSubscribe(1,ADL_AD_SIZE_UNDEF);
        wm_sprintf( RspStr, "adl_adSubscribe: %d", Cell_Handle);
        TRACE((2,RspStr));
        if (Cell_Handle >= OK)
        {
                ret = adl_adDelete (Cell_Handle);
                wm_sprintf( RspStr, "adl_adDelete: %d", ret);
                TRACE((2,RspStr));
                if ( ret != OK) adl_adUnsubscribe (Cell_Handle);
        }
        ret = adl_adRecompact((s32) NULL);
        wm_sprintf( RspStr, "adl_adRecompact: %d", ret);
        TRACE((2,RspStr));
}

void SmsCtrlHandler(u8 Event, u16 Nb)
{
        if(Event==ADL_SMS_EVENT_SENDING_OK)
        {
                if(aplikacja_pobrana)
                ret = adl_adInstall(Cell_Handle);
                else
                { //aktualizacja zakończona bledem wiec restart
                        adl_atCmdCreate ( "AT+CFUN=1" , ADL_AT_PORT_TYPE(ADL_AT_UART1, FALSE), (adl_atRspHandler_t *)
NULL, NULL , NULL);
                }
        }

}

bool Sms_handler(ascii * SmsTel, ascii * SmsTimeLength, ascii * SmsText){
```

Listing 1. c.d.

```
        ascii tekst[0x25];
        wm_sprintf (tekst," Numer telefonu: %s \n\r",SmsTel );
        adl_atSendResponse ( ADL_AT_UNS, tekst );
        wm_sprintf (tekst," Plik do pobrania: %s \n\r",SmsText );
        adl_atSendResponse ( ADL_AT_UNS, tekst );
        if(!zajety)
        {
                wm_strcpy (filename,SmsText);
                wm_strcpy (nr_tel,SmsTel);
                adl_atSendResponse ( ADL_AT_UNS, "przygotowanie A&D...\r\n" );
                init_AD_memory();
        }
        return FALSE;
}

static void evh_sim( u8 event)
{
        if( event != ADL_SIM_EVENT_FULL_INIT ) return;
        handSms = adl_smsSubscribe(Sms_handler, SmsCtrlHandler, Mode);
}

void adl_main ( adl_InitType_e InitType )
{
        adl_atSendResponse( ADL_AT_RSP, "Aplikacja pierwotna FTP...\r\n");
        switch( InitType )
        {
                case ADL_INIT_POWER_ON:
                        adl_atSendResponse( ADL_AT_UNS, "ADL_INIT_POWER_ON\r\n" );
                        break;
                case ADL_INIT_REBOOT_FROM_EXCEPTION:
                        adl_atSendResponse( ADL_AT_UNS, "ADL_INIT_REBOOT_FROM_EXCEPTION\r\n" );
                        break;
                case ADL_INIT_DOWNLOAD_SUCCESS:
                        adl_atSendResponse( ADL_AT_UNS, "ADL_INIT_DOWNLOAD_SUCCESS\r\n" );
                        break;
                case ADL_INIT_DOWNLOAD_ERROR:
                        adl_atSendResponse( ADL_AT_UNS, "ADL_INIT_DOWNLOAD_ERROR\r\n" );
                        break;
                default:
                        adl_atSendResponse( ADL_AT_UNS, "ADL_MAIN_START_REMOTE_MODE\r\n" );
                        break;
        }
        wip_netInit();
        adl_simSubscribe( evh_sim, GPRS_PINCODE);
}
```

R E K L A M A