JTAG-lock-pick 1.x.x
Manual PL 2.0 / 120229

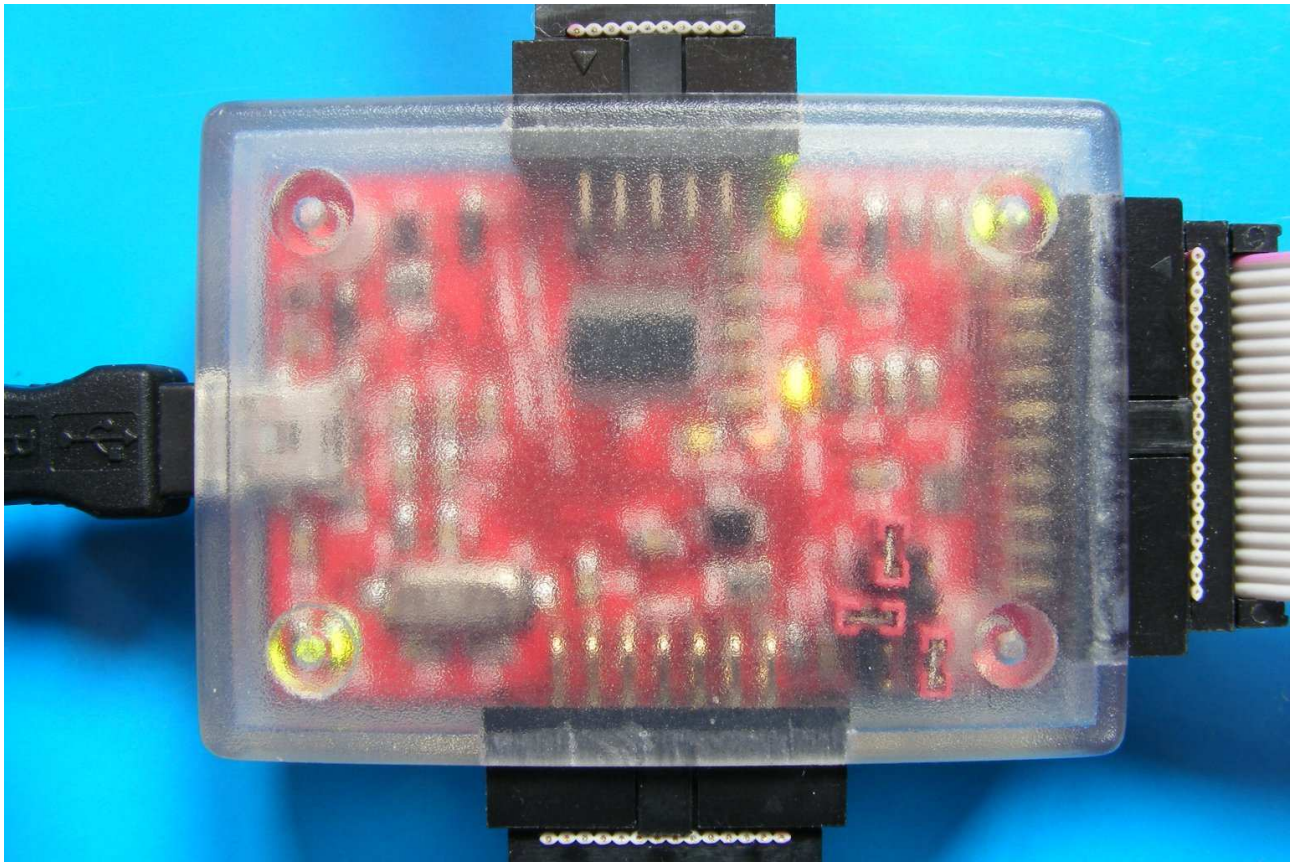
www.distortec.pl
www.freddiechopin.info

Table of contents

1. Wstęp....3
1.1. Obsługiwane układy....5
1.2. „Mocne strony” projektu JTAG-lock-pick....5
1.3. Zawartość zestawu....6
2. Hardware....7
2.1. Złącza....7
2.1.1. JTAG....7
2.1.2. RS-232....7
2.1.3. UART....9
2.2. Diody LED....10
2.3. Zworki....10
2.3.1. Zworka J1 – JVREF....10
2.3.2. Zworka J2 – łączenie JVREF i UVREF....10
2.3.3. Zworka J3 – UVREF....10
3. Sterowniki....11
3.1. libusb-win32....11
3.2. ftd2xx....12
3.3. Zmiana sterownika....12
4. Software....14
4.1. OpenOCD....14
4.2. Atollic TrueSTUDIO....15
4.3. CooCox CoIDE....16
4.4. IAR Embedded Workbench for ARM....17
4.4.1. OpenOCD....17
4.4.2. CooCox CoIARPlugin....17
4.5. Keil MDK-ARM....17
4.6. Rowley CrossWorks for ARM....18
5. Źródła dodatkowych informacji....19
6. Troubleshooting....20
7. Manual changelog....21
8. Hardware changelog....22
9. Appendix....23
9.1. Schemat układu (wersja 1.1)....24
9.2. Lista elementów (bill of materials) (wersja 1.1)....27

1. Wstęp

JTAG-lock-pick to *JTAG* do procesorów z rdzeniem *ARM* podłączany do komputera *PC* przez magistralę *USB*. Urządzenie oparte jest o układ *FTDI FT2232*[1], czyli dwukanałowy konwerter *USB <=> UART/FIFO*. Zaprojektowany układ na pierwszym kanale udostępnia interfejs *JTAG*, natomiast na drugim pełny interfejs *UART / RS-232* (*RXD*, *TXD*, *RTS*, *CTS*, *DCD*, *DSR*, *DTR*, *RI*) z możliwością dołączenia konwertera *RS-485*. Projekt **JTAG-lock-pick** tworzony był jako rozwiązanie bezkompromisowe i rozbudowane, wykorzystujące wszystkie możliwości zastosowanych układów.



Rys. 1: *JTAG-lock-pick - układ w obudowie*

Interfejs *JTAG* pozwala na osiągnięcie częstotliwości zegara do *6MHz*. Dzięki wykorzystaniu magistrali *USB* urządzenie można podłączyć do praktycznie każdego istniejącego komputera *PC*, co nie jest możliwe przy wykorzystaniu interfejsu równoległego (*LPT*), który odchodzi już w całkowite zapomnienie. Dzięki zastosowaniu zaawansowanych układów buforujących z serii *74LVC* możliwa jest komunikacja z układami zasilanymi napięciami z bardzo szerokiego zakresu – od około *1,6V* do *5,5V*.

Jednym z założeń projektu było udostępnienie zarówno interfejsu *UART* jak i *RS-232*. Wybór pomiędzy interfejsem *RS-232* a *UART* realizowany jest automatycznie na podstawie obecności napięcia zasilającego układy buforujące interfejsu *UART* – jeśli napięcie to jest wyższe niż około *1,7V* aktywowany jest interfejs *UART*. Cztery najważniejsze linie interfejsu *UART* (*RXD*, *TXD*, *RTS*, *CTS*) są również buforowane przez układy serii *74LVC*. Pozostałe linie są sterowane bezpośrednio przez układ *FT2232* – sygnały wyjściowe mają poziom *5V*, natomiast próg *„logicznej jedynki”* dla

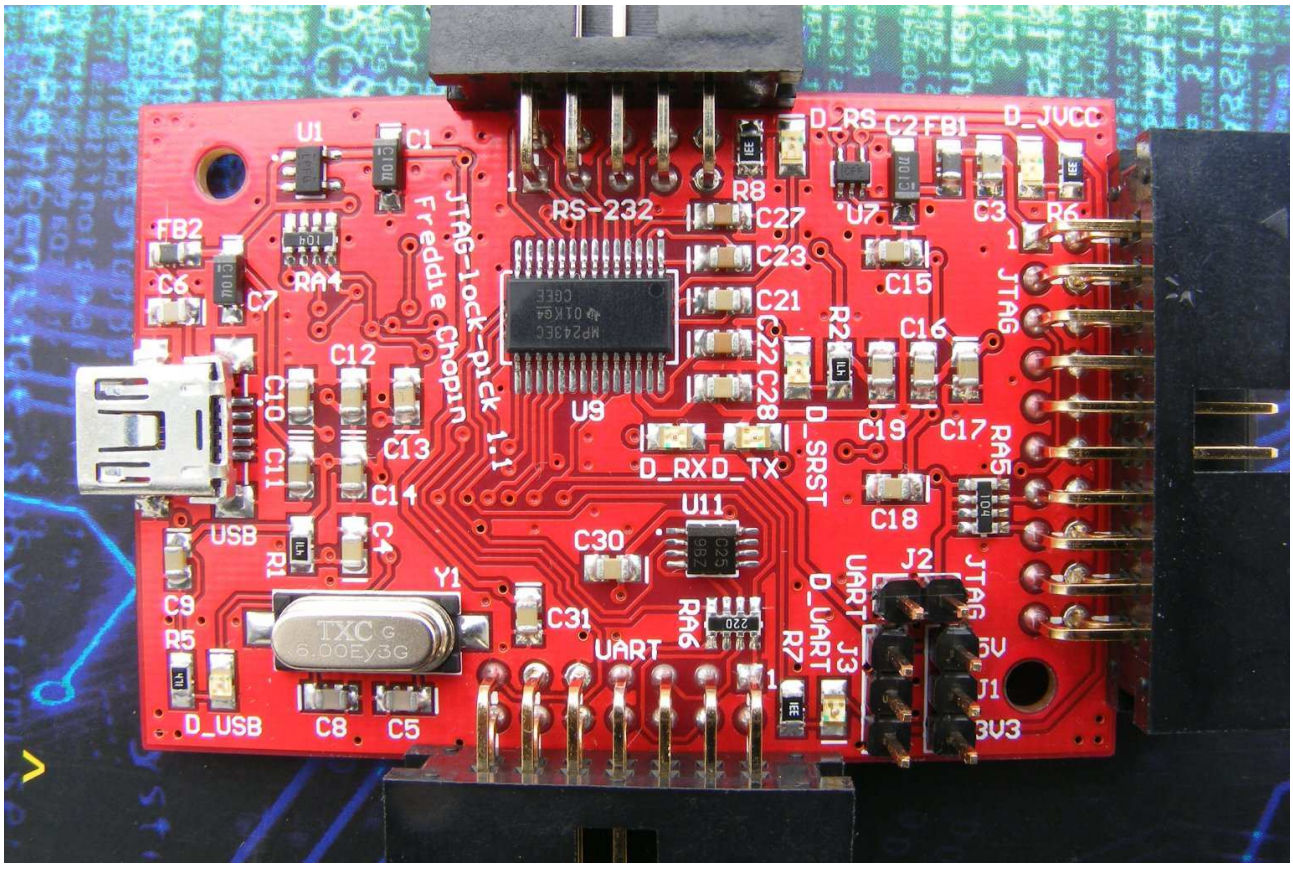
1 http://www.ftdichip.com/Products/ICs/FT2232D.htm

sygnałów wejściowych wynosi (typowo) około *1,6V.* Dzięki doprowadzeniu do gniazda *UART* sygnału *TXDEN* możliwe jest dołączenie do całego układu konwertera *RS-485*.

Zaprojektowany układ umożliwia zasilanie układów buforujących – a więc i układu docelowego (poprzez gniazda *JTAG / UART*) – napięciem *3,3V* lub *5V.* Szyny zasilające bufory odpowiedzialne za *UART* i *JTAG* są od siebie rozdzielone, dodatkowa zworka umożliwia zwarcie ich ze sobą, dzięki czemu możliwe są różne konfiguracje napięciowe. Źródłem tych napięć jest szyna *USB*, maksymalny pobierany przez urządzenie prąd wynosi *500mA* dla napięcia *5V* (wg specyfikacji) lub *150mA* dla napięcia *3,3V,* co jest wartością wystarczającą dla rozbudowanych urządzeń cyfrowych bez układów mocy.

JTAG-lock-pick jest programowo w pełni kompatybilny z interfejsem *JTAGkey*[2] firmy *Amontec*, dzięki czemu w wielu aplikacjach można korzystać z gotowych ustawień konfiguracyjnych zamiast tworzyć je ręcznie. Możliwe jest również wykorzystanie **JTAG-lock-pick** do programowania układów *FPGA*, *CPLD* oraz *AVR* plikami typu *SVF* przy pomocy oprogramowania udostępnianego przez firmę *Amontec*.

JTAG-lock-pick posiada rozdzielone linie *SRST* i *TRST,* które można dodatkowo niezależnie konfigurować na pracę w trybie *push-pull* lub *open-drain*.



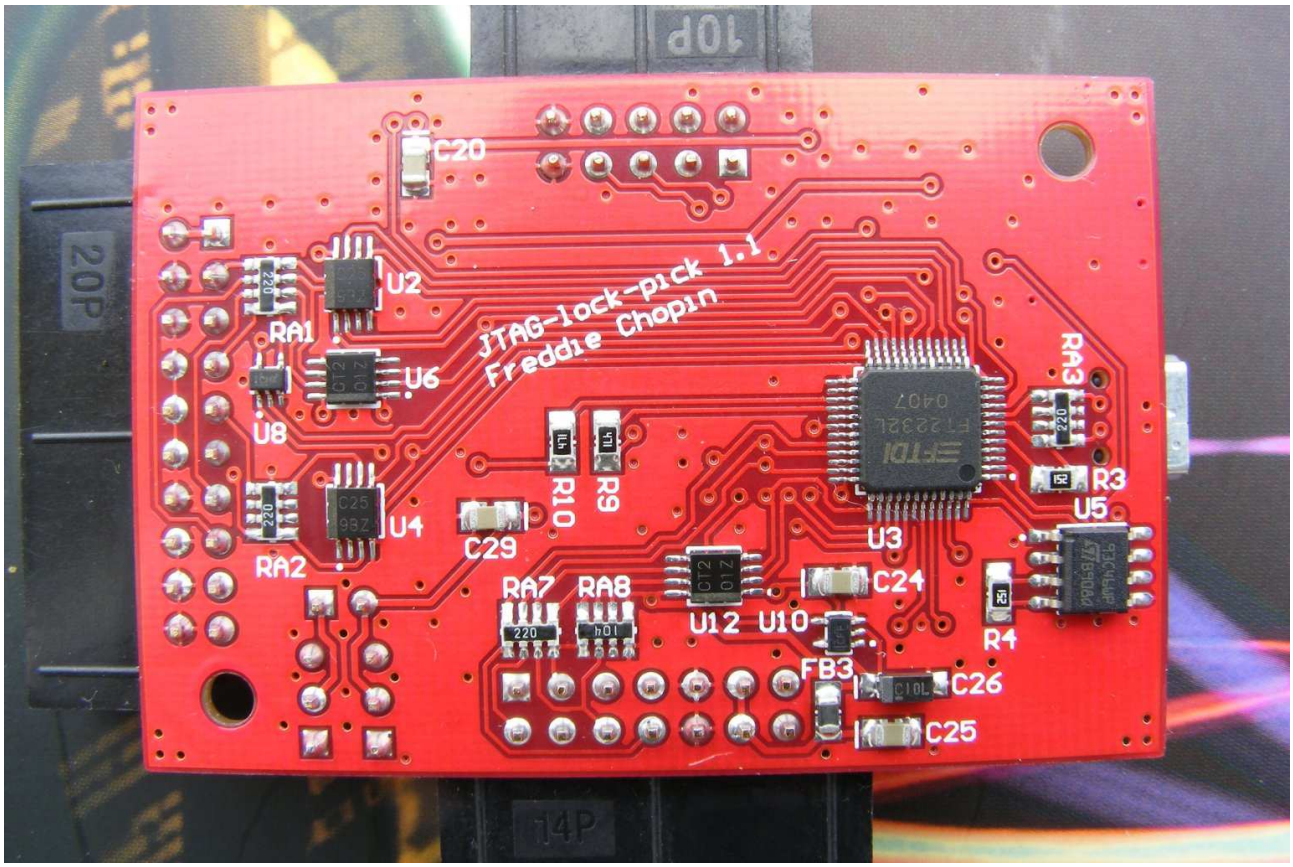
Rys. 2: *JTAG-lock-pick - widok na stronę TOP*

Projekt **JTAG-lock-pick** jest następcą projektu **4R|\/|-JT4G Rev02**[3], eliminującym kilka jego wad. Przede wszystkim poprzednik miał zwarte linie *SRST* i *TRST „z założenia”*, co ograniczało możliwości *debuggowania*. Dodatkowo zastosowany poprzednio układ buforujący *74VHC244* miał istotną wadę, która objawiała się przebiciem napięć wejściowych przez diody zabezpieczające jego wejścia na niepodłączoną linię zasilania, co było bardzo mylące, choć całkowicie bezpieczne dla

2 http://www.amontec.com/jtagkey.shtml
3 http://www.freddiechopin.info/index.php/pl/projekty/49-arm-jtag-rev01-rev02

JTAGa jak i układu docelowego. Obydwa te problemy nie są obecne w projekcie **JTAG-lock-pick**.

Na **rysunku 1** przedstawiona jest całość zamknięta w obudowie, natomiast na **rysunkach 2** oraz **3** przedstawiony jest zmontowany układ **JTAG-lock-pick**.



Rys. 3: *JTAG-lock-pick - widok na stronę BOTTOM*

1.1. Obsługiwane układy

Interfejs **JTAG-lock-pick** i obsługujące go oprogramowanie na komputer *PC* (m.in. *OpenOCD*, *Atollic TrueSTUDIO*, *CooCox CoIDE*, *Keil MDK-ARM*, *IAR Embedded Workbench for ARM*, *Rowley CrossWorks for ARM*, patrz **rozdział 4**) są w stanie komunikować się z praktycznie każdym istniejącym typem procesora *ARM*, w tym z najpopularniejszymi:

- *ARM7* (*LPC2xxx*, *AT91SAM7*, *STR7xx*, ...),
- *ARM9* (*LPC3xxx*, *AT91SAM9*, *STR9xx*, ...),
- *Cortex-M3* (*STM32*, *LM3S*, *LPC17xx*, *AT91SAM3*, …),
- ...

Jedynym czynnikiem limitującym jest obsługa danego procesora przez oprogramowanie na komputerze *PC*.

1.2. „Mocne strony” projektu JTAG-lock-pick

- bezpieczna i bezproblemowa współpraca z układami zasilanymi napięciami z zakresu od około *1,6V* do *5,5V*, sygnały buforowane zaawansowanymi układami z serii *74LVC*,
- interfejsy *UART* i *RS-232*, każdy z kompletem ośmiu standardowych sygnałów, cztery naj-

ważniejsze linie interfejsu *UART* buforowane (około *1,6V* do *5,5V*),
- możliwość zasilania układu docelowego napięciem *3,3V* lub *5V* za pomocą *JTAGa*,
- rozdzielone szyny zasilania buforów *JTAG* i *UART*,
- całość zamknięta w obudowie, dzięki czemu *JTAG* oraz podłączony do niego komputer i układ docelowy są zabezpieczone przed przypadkowym uszkodzeniem,
- możliwość podłączenia do układu konwertera *RS-485*,
- …

1.3. Zawartość zestawu

- *debugger* / programator **JTAG-lock-pick** (*JTAG + RS-232 + UART*), lutowany maszynowo, przetestowany, zamknięty w przezroczystej obudowie (wymiary *24mm x 47mm x 66mm*) z otworami na gniazda wycinanymi na frezarce numerycznej oraz z gumowymi nóżkami, (**rysunek 1**, **2** i **3**),
- taśma przyłączeniowa *JTAG <=> target*, *20cm*, (**rysunek 4**),
- taśma-przejściówka *IDC-10 <=> DB-9* (*RS-232*), *20cm*, (**rysunek 5**),
- dwie wtyczki *IDC-14* zaciskane na taśmie + *20cm* taśmy płaskiej *14-to* żyłowej do stworzenia dowolnego przewodu do interfejsu *UART*,
- kabel *USB mini-B*, czarny, *1,8m*,
- płyta *DVD* z instrukcją, sterownikami, kompletem przydatnych programów – darmowych (*toolchain gcc CodeSourcery*, *IDE Eclipse*, *OpenOCD*, *CooCox CoIDE*) i komercyjnych w wersjach testowych (*Atollic TrueSTUDIO for STM32 Lite*, *Keil MDK-ARM + CooCox CoMDKPlugin*, *IAR Embedded Workbench for ARM + CooCox CoIARPlugin*, *Rowley CrossWorks for ARM*) – wszystkie w najnowszych wersjach.

2. Hardware

2.1. Złącza

2.1.1. JTAG

Rozkład sygnałów w gnieździe *JTAG* (przedstawiony w **tabeli 1**) jest zgodny z tzw. *„standardem”*.

Tabela 1: *Pinologia gniazda JTAG*

JVREF – 1	*2 – JVREF*
nTRST – 3	*4 – GND*
TDI – 5	*6 – GND*
TMS – 7	*8 – GND*
TCK – 9	*10 – GND*
n/c – 11	*12 – GND*
TDO – 13	*14 – GND*
nSRST – 15	*16 – GND*
n/c – 17	*18 – GND*
n/c – 19	*20 – GND*

Linie *JVREF* są liniami zasilającymi bufory (lub docelowy układ, jeśli wybrano zasilanie buforów ze strony komputera *PC*). Na **rysunku 4** pokazana została typowa *20-to* żyłowa taśma połączeniowa.

2.1.2. RS-232

Rozkład sygnałów w gnieździe *RS-232* (przedstawiony w **tabeli 2**) zgodny jest z kolejnością standardową w złączu *DB-9*. Zastosowanie taśmy z zaciskanym męskim gniazdem *DB-9* (przedstawionej na **rysunku 5**) pozwala na uzyskanie podłączanego do **JTAG-lock-pick** standardowego portu szeregowego.

Tabela 2: *Pinologia gniazda RS-232*

DCD – 1	*2 – DSR*
RXD – 3	*4 – RTS*
TXD – 5	*6 – CTS*
DTR – 7	*8 – RI*
GND – 9	*10 – GND*



Rys. 4: *Standardowa taśma połączeniowa JTAG*



Rys. 5: *Taśma połączeniowa RS-232*

2.1.3. UART

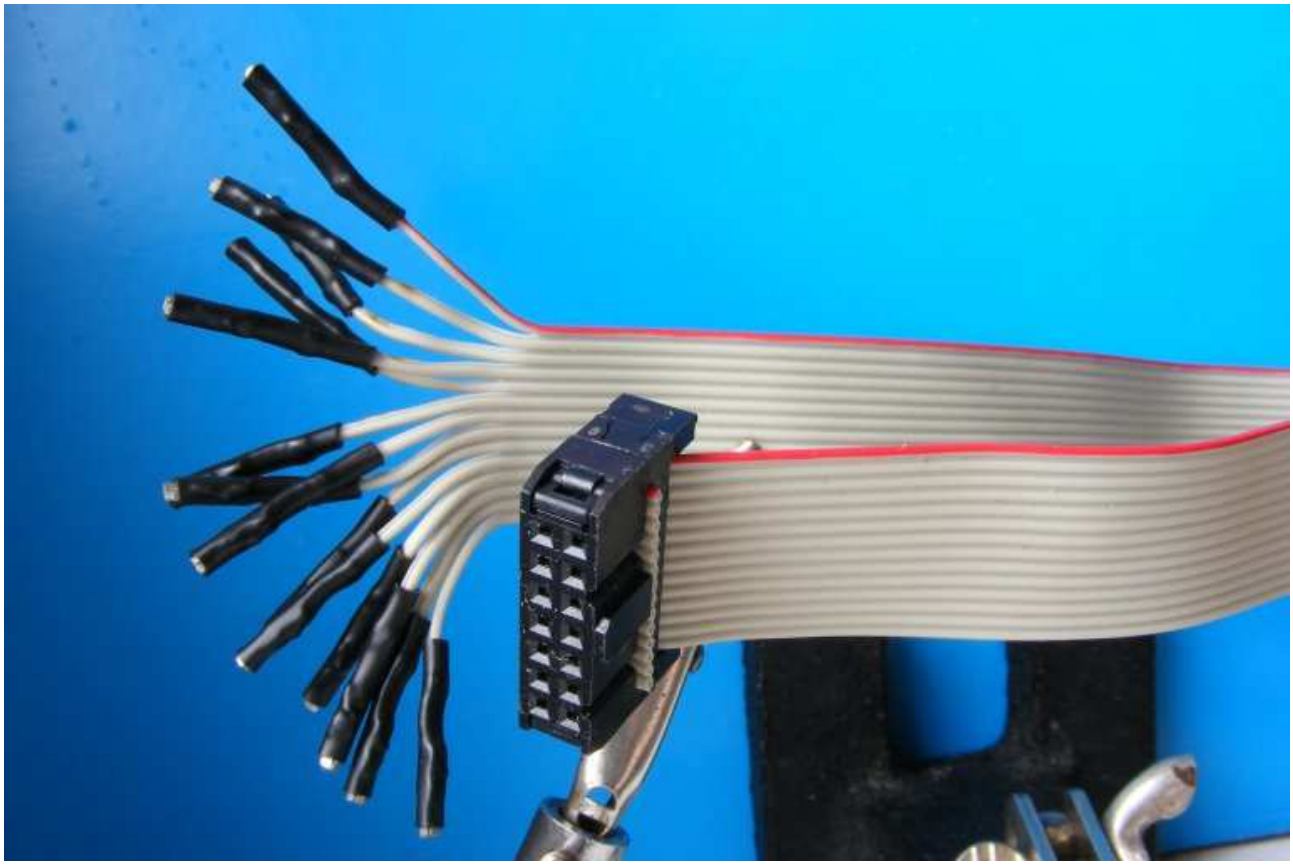
Rozkład sygnałów w gnieździe *UART* (przedstawiony w **tabeli 3**) jest częściowo zgodny z gniazdem *RS-232*.

Tabela 3: *Pinologia gniazda UART*

DCD – 1	*2 – DSR*
RXD – 3	*4 – RTS*
TXD – 5	*6 – CTS*
DTR – 7	*8 – RI*
GND – 9	*10 – GND*
UVREF – 11	*12 – UVREF*
TXDEN – 13	*14 – n/c*

Linie *UVREF* są liniami zasilającymi bufory (lub docelowy układ, jeśli wybrano zasilanie buforów ze strony komputera *PC*).

Interfejs *UART* z **JTAG-lock-pick** można połączyć z układem docelowym za pomocą prostej taśmy *14-to* żyłowej (o konstrukcji analogicznej do tej przedstawionej na **rysunku 4**), taśmy z zakończeniem typu *„flying-leads”* (przykład takiej taśmy przedstawiony został na **rysunku 6**) lub innej – dostosowanej do konkretnego zastosowania.



Rys. 6: *Przykład taśmy UART z zakończeniem "flying-leads"*

Dzięki obecności w gnieździe linii *TXDEN*, do gniazda *UART* można podłączyć typowy *transceiver RS-485* (np. układ *MAX485*) – w takim wypadku linia ta steruje kierunkiem transmisji.

2.2. Diody LED

Na płytce **JTAG-lock-pick** znajduje się siedem diod sygnalizacyjnych *LED*. Ich znaczenie jest następujące:

- *D_USB* (zielona), przy gnieździe *USB* – zakończenie enumeracji urządzenia *USB*, poprawne wykrycie **JTAG-lock-pick** przez komputer *PC*,
- *D_JVCC* (zielona), przy gnieździe *JTAG* – obecność prawidłowego napięcia zasilającego bufory interfejsu *JTAG* (i układ docelowy),
- *D_SRST* (żółta), na środku modułu – linia *nSRST* jest w stanie aktywnym (stan niski) – układ docelowy jest zresetowany,
- *D_RS* (zielona), przy gnieździe *RS-232* – brak prawidłowego napięcia zasilającego bufory interfejsu *UART*, aktywny port *RS-232*, port *UART* nieaktywny,
- *D_UART* (zielona), przy gnieździe *UART* – obecność prawidłowego napięcia zasilającego bufory *UART*, aktywny port *UART*, port *RS-232* nieaktywny,
- *D_RX* (żółta), na środku modułu – odbieranie znaku przez interfejs *UART / RS-232*,
- *D_TX* (czerwona), na środku modułu – nadawanie znaku przez interfejs *UART / RS-232*.

2.3. Zworki

Na płytce **JTAG-lock-pick** znajdują się trzy zworki, służące do konfiguracji zasilania. **Nieprawidłowa konfiguracja zworek może doprowadzić do uszkodzenia *JTAGa*, układu docelowego lub portu *USB* w komputerze *PC*!** Domyślnie wszystkie zworki powinny być zdjęte.

2.3.1. Zworka J1 – JVREF

Założenie zworki w pozycję oznaczoną jako *5V* lub *3V3* powoduje zasilenie buforów *JTAGa* (oraz układu docelowego za pośrednictwem linii *JVREF*) napięciem (odpowiednio) *5V* lub *3,3V*. Umożliwia to *debuggowanie* układów bez doprowadzania do *JTAGa* napięcia zasilającego bufory i / lub zasilanie układu docelowego za pośrednictwem *JTAGa*. **Jeśli układ docelowy ma własne zasilanie, które doprowadzone jest liniami *JVREF*, zworka musi być bezwzględnie zdjęta!**

2.3.2. Zworka J2 – łączenie JVREF i UVREF

Założenie tej zworki powoduje połączenie linii *JVREF* i *UVREF*, dzięki czemu można m.in. zasilać bufory *UART* napięciem doprowadzonym liniami *JVREF*. **Jeśli do *JTAGa* zewnętrznie doprowadzone jest zarówno napięcie *JVREF* jak i *UVREF*, nie wolno zakładać tej zworki!**

2.3.3. Zworka J3 – UVREF

Działanie tej zworki jest identyczne jak zworki *J1*, dotyczy jednak doprowadzonego liniami *UVREF* napięcia zasilającego bufory *UART*.

3. Sterowniki

Pierwsze podłączenie *JTAGa* do komputera zaowocuje znalezieniem nowych urządzeń *USB*, dla których należy zainstalować wybrane sterowniki. Zasadniczo dostępne są dwie opcje sterowników:

- *opensource'owe* sterowniki *libusb-win32*,
- sterowniki producenta układu *FT2232* oparte o bibliotekę *ftd2xx*.

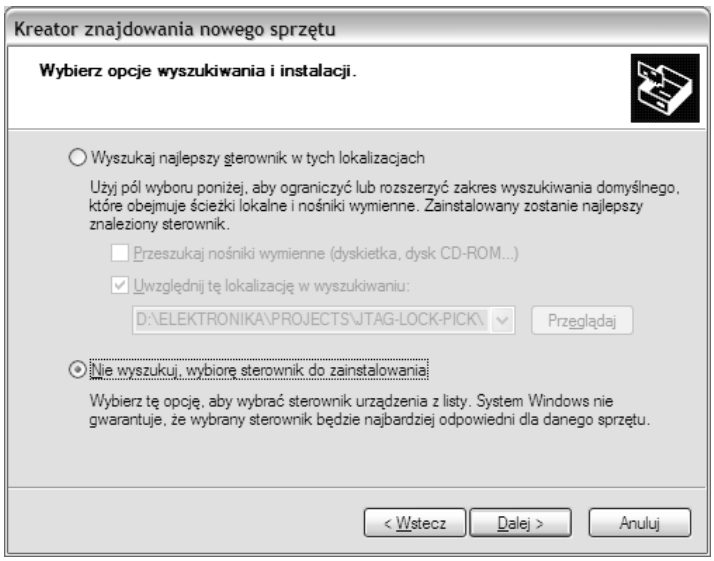
Wybór uzależniony jest od oprogramowania które będzie używane do *debuggowania* przy użyciu **JTAG-lock-pick**. Obecnie dostępne w internecie wersje *OpenOCD* (patrz **rozdział 4.1**) korzystają z biblioteki *libftdi*[4], a więc wymagają sterowników *libusb-win32*. Możliwa (choć skomplikowana) jest również samodzielna kompilacja *OpenOCD* tak aby wykorzystywało ono nieco szybsze (w systemie *Windows*) sterowniki *ftd2xx*. Środowisko *CooCox CoIDE* (patrz **rozdział 4.3**) oraz wtyczki *CooCox CoMDKPlugin* (dla *Keil MDK-ARM*, patrz **rozdział 4.5**) i *CooCox CoIARPlugin* (dla *IAR Embedded Workbench for ARM*, patrz **rozdział 4.4.2**) korzystają ze sterowników *ftd2xx*. Środowisko *Rowley CrossWorks for ARM* (patrz **rozdział 4.6**) potrafi wykorzystać obydwa rodzaje sterowników.

Archiwum zawierające obydwie wersje sterowników dla **JTAG-lock-pick** znajduje się na dołączonej do zestawu płycie *DVD* w katalogu *Drivers*. Najnowsze wersje sterowników można znaleźć też na stronie firmy *DISTORTEC*[5] w dziale *Download* oraz na stronie *Freddiego Chopina*[6] w dziale *Download → Projekty → JTAG-lock-pick*.

3.1. libusb-win32

Opensource'owe sterowniki dotyczą jedynie pierwszego kanału układu *FT2232* – tego który obsługuje interfejs *JTAG*. W przypadku kanału dotyczącego interfejsu *UART / RS-232* należy zainstalować sterowniki *ftd2xx* (patrz **rozdział 3.2**).

Zalecana jest instalacja tego sterownika w trybie całkowicie manualnym, w którym plik sterownika jest wskazywany wprost, a nie wyszukiwany przez system operacyjny – w trakcie instalacji jest to opcja *„Nie wyszukuj, wybiorę sterownik do zainstalowania”* (patrz **rysunek 7**). Ręczna instalacja jest też konieczna w przypadku zmiany sterowników (patrz **rozdział 3.3**).



***Rys. 7:** Ręczna instalacja sterowników*

Prawidłowa instalacja sterownika sygnalizowana jest przez pojawienie się elementu *JTAG-lock-pick – USB <=> JTAG adapter* w grupie *libusb-win32 devices* w *Menadżerze urządzeń*. Sytuacja ta (po zainstalowaniu sterowników *ftd2xx* dla interfejsu *UART / RS-232*) przedstawiona jest na **rysunku 8**.

4 Udostępnianie *OpenOCD* skompilowanego do użycia z *„zamkniętymi”* sterownikami *ftd2xx* rzekomo narusza licencję *GPLv2*

5 http://www.distortec.pl/

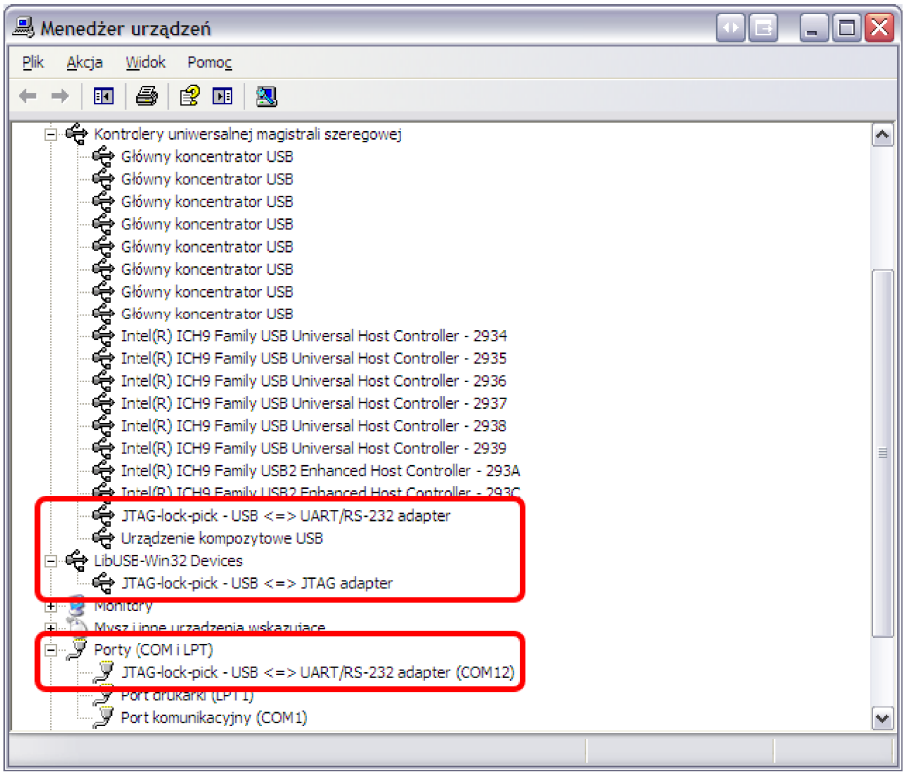
6 http://www.freddiechopin.info/

3.2. ftd2xx

Po zainstalowaniu sterowników, w *Menadżerze Urządzeń* systemu powinny pojawić się następujące cztery elementy:

- w grupie K*ontrolery uniwersalnej magistrali szeregowej*: *JTAG-lock-pick - USB <=> JTAG adapter* (jedynie jeśli dla kanału *JTAGa* zainstalowano sterowniki *ftd2xx* a nie *libusb-win32*), *JTAG-lock-pick - USB <=> UART/RS-232 adapter* oraz *Urządzenie kompozytowe USB*;
- w grupie *Porty COM i LPT*: *JTAG-lock-pick - USB <=> UART/RS-232 adapter (COM xx)*;

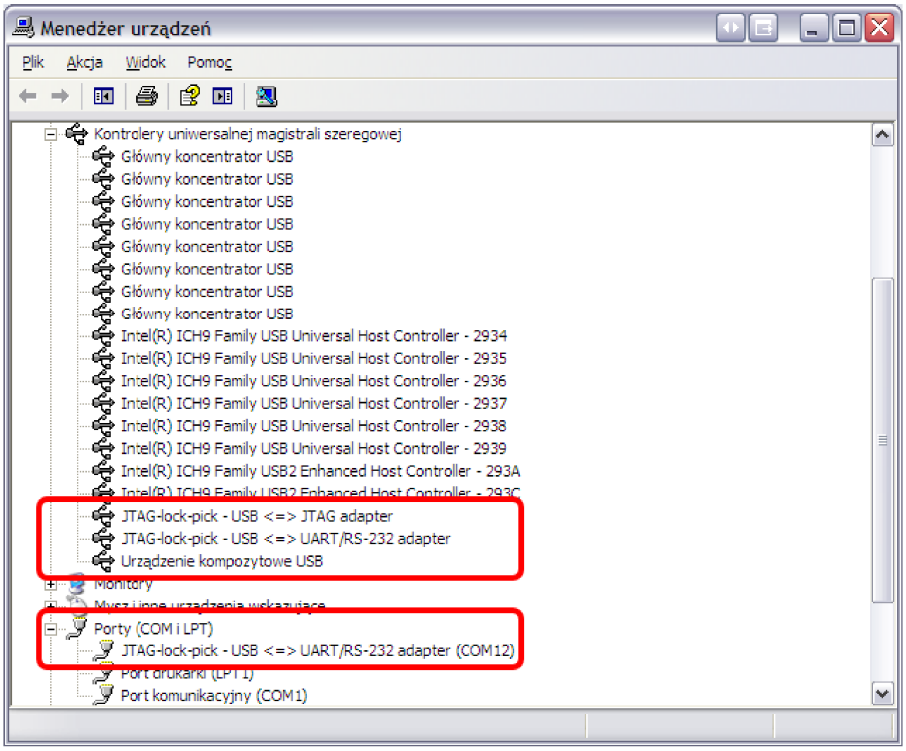
Zawartość *Menadżera Urządzeń* po zainstalowaniu kompletu sterowników *ftd2xx* przedstawiona jest na **rysunku 9**.



Rys. 8: *Zainstalowane sterowniki libusb-win32 (JTAG) oraz ftd2xx (UART / RS-232)*

3.3. Zmiana sterownika

Jeśli zajdzie potrzeba zmiany sterownika dla kanału *JTAGa* (np. ze względu na błąd lub zmianę używanej aplikacji), należy w *Menadżerze Urządzeń* odnaleźć pozycję *JTAG-lock-pick - USB <=> JTAG adapter* i dwukrotnie klikając ją otworzyć jej właściwości. Następnie w zakładce *Sterownik* należy użyć przycisku *Aktualizuj sterownik...*, co spowoduje otworzenie kreatora aktualizacji sterowników. Pamiętać należy o tym, aby nie pozwolić systemowi na automatyczne wyszukiwanie sterowników – skorzystać należy z opcji *„Nie wyszukuj, wybiorę sterownik do zainstalowania”* (patrz **rysunek 7**) – gdyż w przeciwnym wypadku system z dużym prawdopodobieństwem zainstaluje ponownie tą samą wersję, nie wprowadzając żadnych zmian.



Rys. 9: *Zainstalowane sterowniki ftd2xx*

4. Software

Wśród programów w których można korzystać z **JTAG-lock-pick** znajdują się najpopularniejsze narzędzia dla procesorów z rdzeniem *ARM*, m.in.: *OpenOCD*, *Atollic TrueSTUDIO*, *CooCox CoIDE*, *Keil MDK-ARM*, *IAR Embedded Workbench for ARM* oraz *Rowley CrossWorks for ARM*. Korzystanie z **JTAG-lock-pick** w tych programach zostało szczegółowo opisane w kolejnych podrozdziałach.

Ze strony oprogramowania **JTAG-lock-pick** może być używany jako *Amontec JTAGkey*. Dzięki temu użytkowanie układu jest bardzo proste – w większości przypadków wykorzystywane oprogramowanie będzie posiadało gotowe konfiguracje dla tego *JTAGa*.

4.1. OpenOCD[7]

Podstawowa komenda uruchamiająca *OpenOCD* powinna wyglądać następująco:

```
> openocd -f interface/jtagkey.cfg -f target/XXX.cfg
```

lub

```
> openocd -f interface/jtagkey.cfg -f board/XXX.cfg
```

(w zależności od tego, gdzie zlokalizowany zostanie plik konfiguracyjny dla docelowego urządzenia / układu).

Dla przykładu – wywołanie *OpenOCD* dla układu *STM32* (*ARM Cortex-M3*) powinno dać efekt zbliżony do poniższego (komunikaty zależne będą oczywiście od wersji *OpenOCD* i używanego układu docelowego):

```
> openocd -f interface/jtagkey.cfg -f target/stm32f1x.cfg
Open On-Chip Debugger 0.5.0 (2011-08-09-23:21)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.berlios.de/doc/doxygen/bugs.html
Info : only one transport option; autoselect 'jtag'
1000 kHz
adapter_nsrst_delay: 100
jtag_ntrst_delay: 100
cortex_m3 reset_config sysresetreq
Info : clock speed 1000 kHz
Info : JTAG tap: stm32.cpu tap/device found: 0x3ba00477 (mfg: 0x23b,
part: 0xba00, ver: 0x3)
Info : JTAG tap: stm32.bs tap/device found: 0x16410041 (mfg: 0x020,
part: 0x6410, ver: 0x1)
Info : stm32.cpu: hardware has 6 breakpoints, 4 watchpoints
```

OpenOCD można również wykorzystywać jedynie do programowania układu docelowego z linii poleceń (lub przy użyciu plików wsadowych / skryptów) przy użyciu komendy:

```
> openocd -f interface/jtagkey.cfg -f target/XXX.cfg -c "init; reset
halt; flash write_image erase YYY.EXT; reset run; shutdown"
```

Komenda ta łączy w sobie kilka poleceń *OpenOCD*:

- *init* – wymagane jest przed kolejnymi komendami *„wykonywalnymi”* (innymi niż konfiguracyjne) w linii poleceń,
- *reset halt* – resetuje i zatrzymuje układ, co jest konieczne przed programowaniem układu,

7 http://openocd.sourceforge.net/

- *flash write_image erase YYY.EXT* – ładuje do układu zawartość z pliku *YYY.EXT* (gdzie *EXT* to rozszerzenie *hex*, *bin* lub *elf*), uprzednio kasując pamięć (jedynie obszary które zostaną zapisane),
- *reset run* – resetuje i uruchamia układ, dzięki czemu startuje wgrany program,
- *shutdown* – zamyka sesję *OpenOCD*.

Przykład programowania układu *STM32* plikiem *stm32_blink_led.hex* przedstawiony jest poniżej.

```
> openocd -f interface/jtagkey.cfg -f target/stm32f1x.cfg -c "init;
reset halt; flash write_image erase stm32_blink_led.hex; reset run;
shutdown"
Open On-Chip Debugger 0.5.0 (2011-08-09-23:21)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.berlios.de/doc/doxygen/bugs.html
Info : only one transport option; autoselect 'jtag'
1000 kHz
adapter_nsrst_delay: 100
jtag_ntrst_delay: 100
cortex_m3 reset_config sysresetreq
Info : clock speed 1000 kHz
Info : JTAG tap: stm32.cpu tap/device found: 0x3ba00477 (mfg: 0x23b,
part: 0xba00, ver: 0x3)
Info : JTAG tap: stm32.bs tap/device found: 0x16410041 (mfg: 0x020,
part: 0x6410, ver: 0x1)
Info : stm32.cpu: hardware has 6 breakpoints, 4 watchpoints
Info : JTAG tap: stm32.cpu tap/device found: 0x3ba00477 (mfg: 0x23b,
part: 0xba00, ver: 0x3)
Info : JTAG tap: stm32.bs tap/device found: 0x16410041 (mfg: 0x020,
part: 0x6410, ver: 0x1)
target state: halted
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x0800024c msp: 0x20000ed0
auto erase enabled
Info : device id = 0x20036410
Info : flash size = 128kbytes
wrote 2048 bytes from file stm32_blink_led.hex in 0.312500s (6.400
KiB/s)
Info : JTAG tap: stm32.cpu tap/device found: 0x3ba00477 (mfg: 0x23b,
part: 0xba00, ver: 0x3)
Info : JTAG tap: stm32.bs tap/device found: 0x16410041 (mfg: 0x020,
part: 0x6410, ver: 0x1)
shutdown command invoked
```

4.2. Atollic TrueSTUDIO[8]

Według informacji na stronie producenta pełna wersja pakietu *Atollic TrueSTUDIO* wspiera interfejsy oparte o układ *FTDI FT2232* (w tym **JTAG-lock-pick** dzięki kompatybilności z *JTAGkey*) w sposób natywny[9]. W darmowej wersji *Lite* dla mikrokontrolerów *STM32* można korzystać z **JTAG-lock-pick** za pośrednictwem *OpenOCD* (patrz **rozdział 4.1**), choć współpraca nie jest bezproblemowa.

Przed przystąpieniem do konfiguracją należy skompilować projekt i upewnić się, że stworzony został plik wynikowy z rozszerzeniem *.elf*. W tle należy uruchomić *OpenOCD – „zewnętrznie”* (z wiersza poleceń systemu operacyjnego) lub konfigurując je w menu *Run* > *External Tools* > *Exter-*

8 http://www.atollic.com/index.php/truestudio
9 http://www.atollic.com/index.php/truestudio/targets/jtagdongles/amontec

nal Tools Configurations..., przykładowa komenda uruchamiająca *OpenOCD* dla *STM32* ma następującą postać:

```
> openocd -f interface/jtagkey.cfg -f target/stm32f1x.cfg -c
"reset_config trst_and_srst"
```

Konfigurację *debuggowania* można przeprowadzić za pomocą opcji *Debug Configurations...* z menu *Run*. W nowo otwartym oknie należy stworzyć nową konfigurację typu *Embedded C/C++ Application* i skonfigurować w odpowiedniej kolejności następujące opcje:

1. w zakładce *Main* w pole *Project* należy wpisać (lub wybrać przyciskiem *Browse...*) projekt którego dotyczyć będzie dana konfiguracja, a następnie w znajdujące się wyżej pole *C/C++ Application* należy wpisać (lub wybrać przyciskiem *Search Project...* lub *Browse...*) plik wynikowy projektu z rozszerzeniem *.elf*; w większości przypadków pola te zostaną ustawione automatycznie przy tworzeniu konfiguracji;
2. w zakładce *Debugger* należy wybrać opcję *Connect to remote GDB server*, a w pole *Port number* wpisać wartość *3333*, pozostałe opcje są ignorowane – ich wartość nie ma znaczenia i nie należy się nimi sugerować;
3. w zakładce *Startup Debug* należy usunąć całą zawartość pola *Initialization Commands* i wpisać tam:

```
# send "reset halt" to OpenOCD
monitor reset halt
# load application to target via GDB
load
# send "reset halt" to OpenOCD
monitor reset halt
# set temporary breakpoint at main() and resume target
tbreak main
continue
```

4. należy zatwierdzić wszystkie zmiany przyciskiem *Apply* i można od razu rozpocząć sesję *debuggowania* klikając przycisk *Debug*.

Problemy we współpracy pakietu *Atollic TrueSTUDIO* w wersji *Lite* z programem *OpenOCD* to m.in.:

- brak możliwości *„zresetowania”* układu docelowego podczas sesji *debuggowania* przyciskiem *Restart*;
- komunikat systemu operacyjnego o nieprawidłowym zamknięciu aplikacji *GDB* podczas kończenia sesji *debuggowania*;

4.3. CooCox CoIDE[10]

W środowisku tym należy skorzystać z gotowej konfiguracji dla *Amontec JTAGkey*. Wszelkie opcje konfiguracyjne dotyczące interfejsu *JTAG* odnaleźć można w menu *Debug* > *Debug Configuration*, po kliknięciu na dostępną tam konfigurację dla danego projektu (zwykle będzie to *nazwa_projektu-.configuration*). Ze znajdującej się na pierwszej zakładce (*Debugger*) w *groupboxie Hardware* listy wyboru *Adapter* należy wybrać pozycję *Amontec-JTAGkey* i po kliknięciu przycisku *Apply*, nacisnąć przycisk *Close*.

10 http://www.coocox.org/CooCox_CoIDE.htm

4.4. IAR Embedded Workbench for ARM[11]

W pakiecie firmy *IAR* możliwe są dwa podejścia do używania interfejsu **JTAG-lock-pick** – można korzystać z *OpenOCD* (patrz **rozdział 4.1**) lub z wtyczki *CooCox CoIARPlugin*.

4.4.1. OpenOCD

Pierwszym krokiem jest oczywiście uruchomienie w tle *OpenOCD* (przez wiersz poleceń systemu operacyjnego) z parametrami odpowiednimi dla danego układu docelowego, zwykle wywołanie będzie wyglądało mniej więcej tak:

```
openocd -f interface/jtagkey.cfg -f target/XXX.cfg
```

Następnie w pakiecie *IAR Embedded Workbench for ARM* w opcjach projektu (*Project > Options*) należy wybrać z bocznego menu pozycję *Debugger.* W zakładce *Setup* z listy *Driver* należy wybrać pozycję *GDB Server,* a w zakładce *Download* zaznaczyć opcję *Use flash loader(s)*. Następnie z bocznego menu należy wybrać pozycję *GDB Server* i w zakładce *GDB Server* w pole *TCP/IP address of hostname [.port]* wpisać *localhost.*

4.4.2. CooCox CoIARPlugin[12]

Po instalacji wtyczki w opcjach projektu (*Project > Options*) należy wybrać z bocznego menu pozycję *Debugger.* W zakładce *Setup* z listy *Driver* należy wybrać pozycję *RDI*, a w zakładce *Download* zaznaczyć opcję *Use flash loader(s)*. Następnie z bocznego menu należy wybrać pozycję *RDI* i w pole *Manufacturer RDI driver* wprowadzić ścieżkę do pliku *CoRDI.dll* znajdującego się w folderze instalacyjnym wtyczki *CooCox CoIARPlugin* (zwykle będzie to *c:\Program Files\CooCox\CoIARPlugin\CoRDI.dll*, w systemie *64-bitowym* użyty będzie zwykle katalog *Program Files (x86)*) oraz zaznaczyć opcję *Allow hardware reset*. Po zamknięciu opcji projektu przyciskiem *OK* w oknie aplikacji pojawi się nowe menu – *RDI*. Należy wybrać jedyną aktywną w tym menu opcję – *Configure* – i w nowo otwartym oknie wybrać z bocznej listy układ docelowy, a z listy *Adapter* w *groupboxie Adapter Config* wybrać pozycję *Amontec-JTAGkey.*

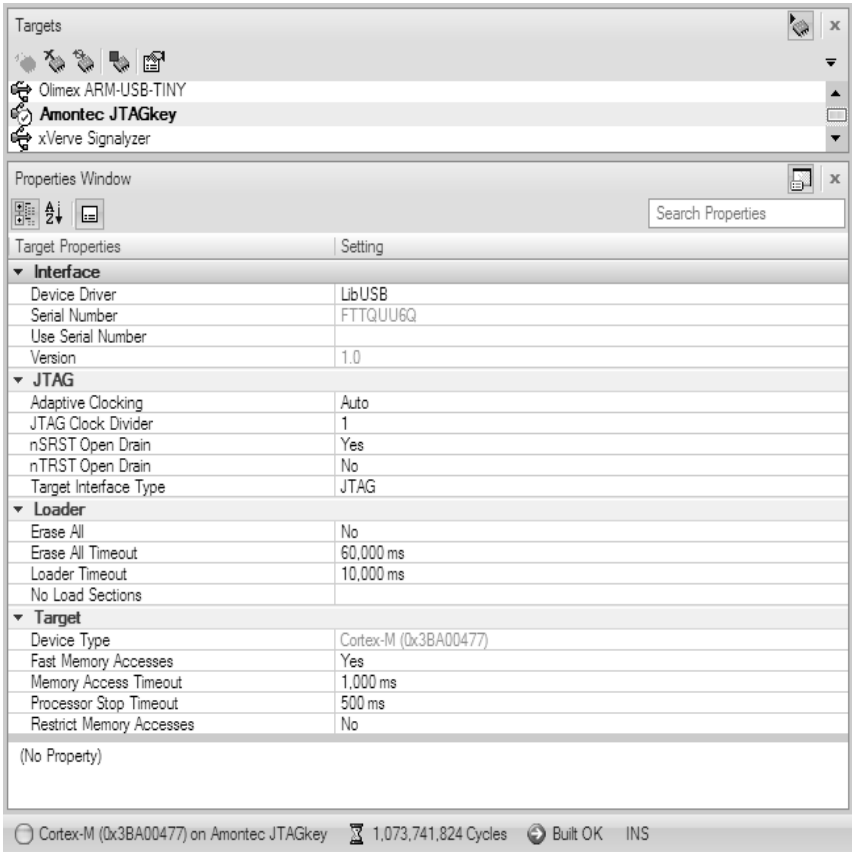
4.5. Keil MDK-ARM[13]

Korzystanie z **JTAG-lock-pick** w środowisku *Keil MDK-ARM* (znanym też pod nazwami *µVision* lub *RealView*) jest możliwe za pomocą wtyczki *CooCox CoMDKPlugin*[14]. Po zainstalowaniu jej, w opcjach projektu (*Project > Options for Target 'nazwa_projektu'*, opcja dostępna tylko po zaznaczeniu projektu w oknie *Projects*) w zakładce *Utilities* należy z listy wyboru pod *Use Target Driver for Flash Programming* wybrać pozycję *CooCox Debugger,* zaznaczyć opcję *Update Target before Debugging,* a następnie nacisnąć przycisk *Settings*. W nowo otwartym oknie należy przejść do zakładki *Debug* i ze znajdującej się w *groupboxie USB Adapter* listy wyboru *Adapter* wybrać pozycję *JTAGkey.* Po naciśnięciu przycisku *OK* i powrocie do opcji projektu należy przejść do zakładki *Debug* i wybrać znajdującą się po prawej stronie opcję *Use*, a z listy wyboru obok wybrać ponownie pozycję *CooCox Debugger* – nie ma potrzeby ponownego ustawiania opcji przyciskiem *Settings*, ponieważ są one współdzielone (zawsze takie same dla zakładki *Utilities* i *Debug*). Poniżej należy jeszcze zaznaczyć opcję *Load Application at Startup* oraz *Run to main()*. Całą konfigurację należy zatwierdzić przyciskiem *OK*.

11 http://www.iar.com/en/Products/IAR-Embedded-Workbench/ARM/
12 http://www.coocox.org/CoLinkGuide/CoIARPlugin.html
13 http://www.keil.com/arm/mdk.asp
14 http://www.coocox.org/CoLinkGuide/CoMDKPlugin.html

4.6. Rowley CrossWorks for ARM[15]

Używanie **JTAG-lock-pick** w środowisku *Rowley CrossWorks for ARM* ogranicza się do dwukrotnego kliknięcia w pozycję *Amontec JTAGkey* w oknie *Targets*. Prawidłowe połączenie z układem sygnalizowane jest na pasku statusu aplikacji oraz wyświetleniem numeru seryjnego *JTAGa* i *Device ID* układu docelowego w oknie *Properties Window* dla *Amontec JTAGkey* – sytuacja taka przedstawiona jest na **rysunku 10**. W niektórych przypadkach może być konieczna modyfikacja parametru *JTAG Clock Divider.*



Rys. 10: *Prawidłowe połączenie z układem w środowisku Rowley CrossWorks for ARM*

15 http://www.rowley.co.uk/arm/index.htm

5. Źródła dodatkowych informacji

Dodatkowe informacje i pomoc dotyczącą *JTAGa*, oprogramowania, *debuggowania* i procesorów *ARM* można znaleźć w wielu miejscach w internecie:

- strona firmy *DISTORTEC* (http://www.distortec.pl/),
- strona *Freddiego Chopina* (http://www.freddiechopin.info/),
- forum *Elektrody* (http://www.elektroda.pl/rtvforum/),
- forum *SparkFun* (http://forum.sparkfun.com/viewforum.php?f=18),
- strona *OpenOCD* (http://openocd.sourceforge.net/),
- strona pakietu *Yagarto* (http://www.yagarto.de/),
- strona pakietu *WinARM* (informacje często nieaktualne!) (http://www.siwawi.arubi.uni-kl.de/avr_projects/arm_projects/),
- *google* (http://www.google.pl/).

6. Troubleshooting

Problem: Po uruchomieniu *OpenOCD* pojawia się komunikat:

```
OpenOCD Error: unable to open ftdi device
```

Geneza: *OpenOCD* do komunikacji z układem *FTDI FT2232* może wykorzystywać dwa różne sterowniki – *„firmowe” ftd2xx* lub *opensource'owe libusb-win32*. Wybór sterownika dokonywany jest już na etapie kompilacji *OpenOCD*. Komunikat taki wskazywać może na kilka problemów:

1. *JTAG* nie jest prawidłowo podłączony do komputera,
2. *JTAG* jest *„zablokowany”* przez inny program lub inną sesję *OpenOCD*,
3. Zainstalowane zostały niewłaściwe sterowniki do *JTAGa*.

Rozwiązanie: Ad 1. Skontrolować połączenie *JTAG <=> PC*.
Ad 2. Zamknąć pozostałe sesje *OpenOCD* lub inne programy połączone z *JTAGiem*.
Ad 3. Odinstalować niewłaściwy i zainstalować właściwy sterownik (patrz **rozdział 3.3**).

Problem: Próba programowania / *debuggowania* układu w pakiecie *CooCox CoIDE* kończy się błędem o treści:

```
Target Chip not found
```

Geneza: Pakiet *CooCox CoIDE* w wersji *1.4.0* (najnowsza w chwili pisania tego dokumentu) nie współpracuje prawidłowo z *JTAGami* opartymi o układ *FTDI FT2232*.

Rozwiązanie: Należy użyć wcześniejszej wersji *1.3.0* lub przetestować nowsze wydania gdy zostaną one udostępnione.

7. Manual changelog

1.0 (19.08.2009)	
1.1 (29.08.2009)	1. Zmiana informacji o lokalizacji sterowników *libusb-win32* – obecnie sterowniki te nie są dołączane do *OpenOCD*, na stronie dostępna jest paczka z obydwoma wersjami sterowników dla **JTAG-lock-pick** 2. Dodana informacja o podforum *OpenOCD* na forum *SparkFun*
1.2 (15.03.2010)	1. Drobne poprawki 2. Dodanie informacji o wersji *1.1* projektu (rozdział Hardware changelog, aktualizacja zdjęć, dodanie schematu i listy elementów) 3. Dodanie dwóch zrzutów ekranu z systemowego *Menadżera Urządzeń* do rozdziału o finalnych sterownikach 4. Aktualizacja zawartości konsoli po wywołaniu *OpenOCD* w wersji *0.4.0*
1.3 (26.12.2010)	1. Drobne poprawki 2. Dodanie informacji o wersji *1.1.1* płytki 3. Opis programowania *EEPROMu* w programie *FT_Prog* 4. Środowisko *Rowley CrossWorks for ARM*: informacja o obsłudze sterowników *libusb-win32*, aktualizacja obrazka 5. Wiele komend przekazywanych do *OpenOCD* poprzez *„-c”* połączono w jeden łańcuch, elementy rozdzielone średnikami
1.3.1 (10.12.2011)	1. Drobne poprawki 2. Doprecyzowanie informacji o tym kiedy trzeba instalować sterowniki *„bazowe”* i programować *EEPROM*
2.0 (29.02.2012)	1. Poprawki większości opisów 2. Usunięcie informacji o elementach i o wersjach *JTAGa* (dostępna jest tylko zmontowana wersja *FULL*) 3. Dodanie obsługiwanych układów, *„mocnych stron”* i zawartości zestawu 4. Usunięcie informacji o sterownikach *„bazowych”* i programowaniu *EEPROMu* (dostępne wersje mają zaprogramowany *EEPROM*) 5. Dodanie informacji o zmianie sterowników 6. Dodanie informacji o innych aplikacjach obsługujących **JTAG-lock-pick** 7. Usunięcie schematu i listy elementów dla wersji *1.0* (dostępna jest tylko wersja *1.1*) 8. Dodane tłumaczenie na angielski

8. Hardware changelog

1.0 (19.02.2009)

1.1 (31.01.2010) Schemat:

1. Drobne poprawki
2. Układ *LP2980* (*50mA max*) zastąpiony układem *LP2985* (*150mA max*) (układy są wzajemnie kompatybilne)
3. Zmiana kolejności diod i rezystorów w dwóch połączeniach (zamiast dioda-rezystor jest rezystor-dioda) – diody *D_UART* i *D_RS* są bardziej odsunięte od gniazd *IDC*, dzięki czemu są lepiej widoczne
4. Wartość drabinki rezystorowej *RA4* zmieniona z *4 x 10kR* na *4 x 100kR*

Płytka *PCB*:

1. Drobne poprawki
2. Usunięcie wszystkich połączeń *„wprost”* między sąsiednimi padami (przy układzie *MAX3243* i przy drabinkach rezystorowych *4 x 100kR*)
3. Dopasowanie mechaniczne płytki do rzeczywistych wymiarów obudowy *Z-24A* (umiejscowienie otworów montażowych, gniazda *IDC-20* oraz diod *D_USB* i *D_JVCC*)
4. Cofnięcie gniazd *IDC „w głąb”* płytki

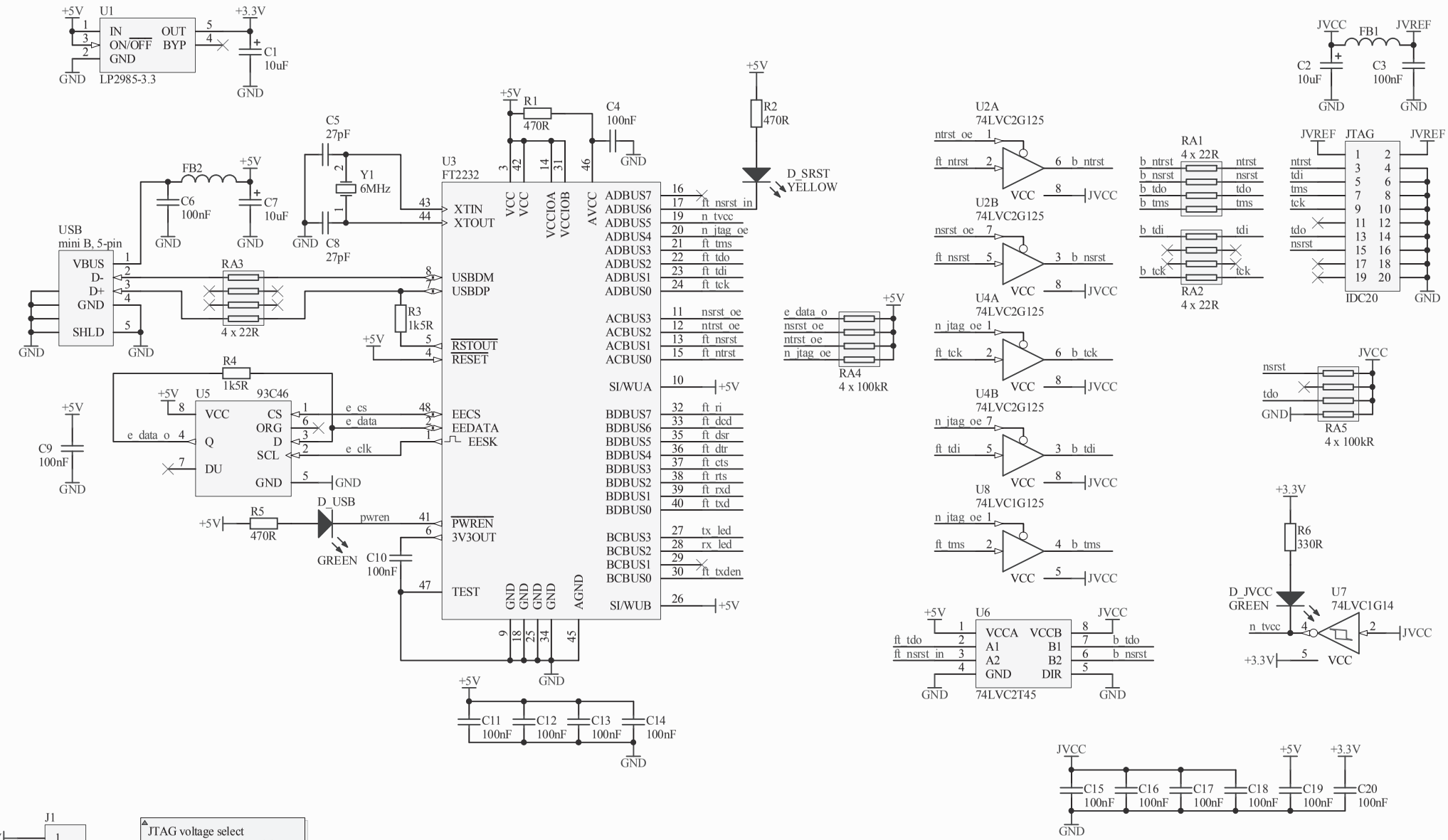
1.1.1 (22.11.2010) Płytka *PCB*:

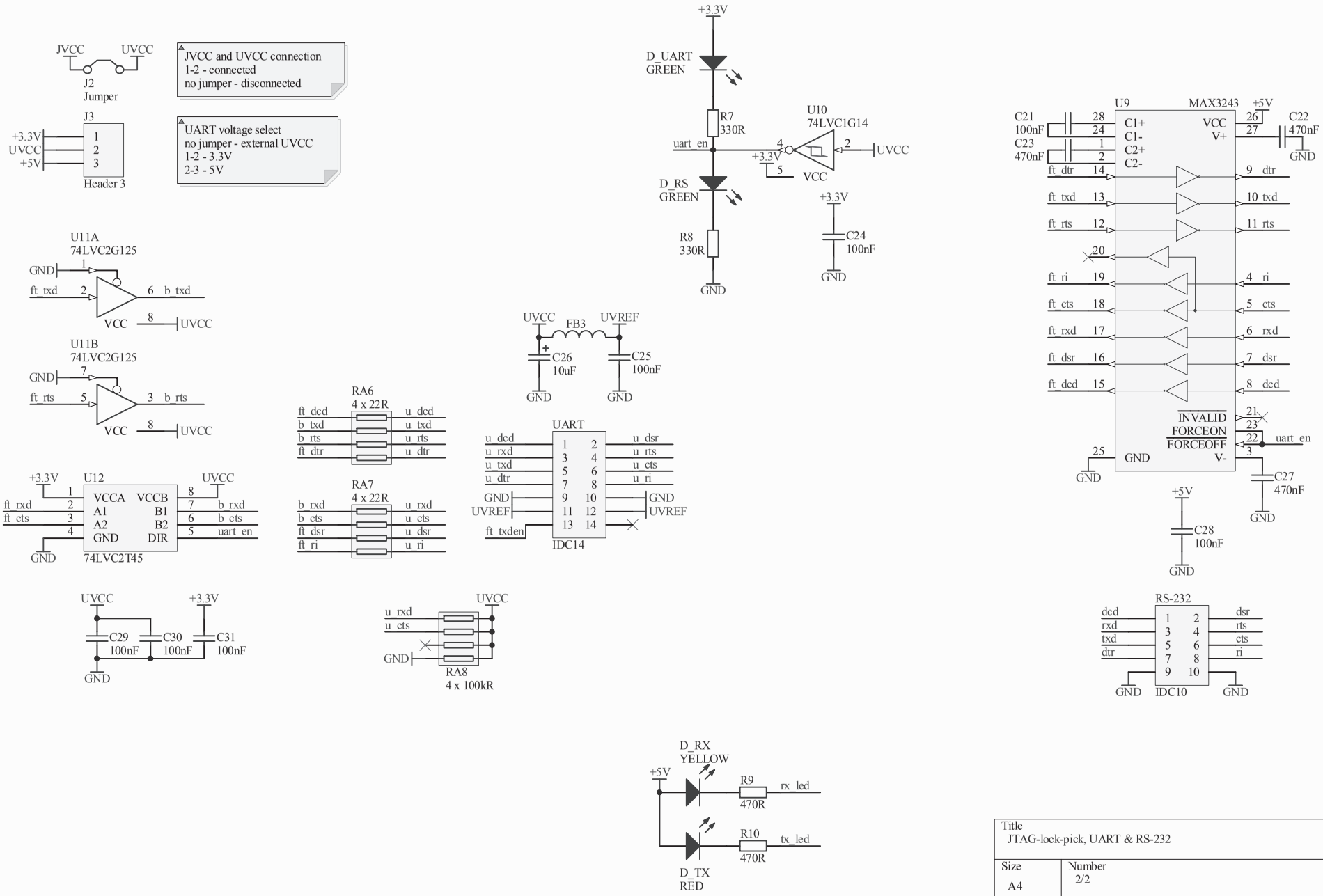
1. Dopasowanie położenia otworów montażowych do opublikowanych (w końcu) dokładnych rysunków technicznych obudowy *Z-24A*

9. Appendix

– schemat układu (wersja *1.1*), str 24
– lista elementów (*bill of materials*) (wersja *1.1*), str 27

9.1. Schemat układu (wersja 1.1)


U1
LP2985-3.3
C1 10uF
C2 10uF
C3 100nF
FB1
JVCC
JVREF
U3
FT2232
XTIN
XTOUT
USBDM
USBDP
RSTOUT
RESET
EECS
EEDATA
EESK
PWREN
3V3OUT
TEST
SI/WUA
SI/WUB
ADBUS7 ft_nsrst_in
ADBUS6 n_tvcc
ADBUS5 n_jtag_oe
ADBUS4 ft_tms
ADBUS3 ft_tdo
ADBUS2 ft_tdi
ADBUS1 ft_tck
ADBUS0
ACBUS3 nsrst_oe
ACBUS2 ntrst_oe
ACBUS1 ft_nsrst
ACBUS0 ft_ntrst
BDBUS7 ft_ri
BDBUS6 ft_dcd
BDBUS5 ft_dsr
BDBUS4 ft_dtr
BDBUS3 ft_cts
BDBUS2 ft_rts
BDBUS1 ft_rxd
BDBUS0 ft_txd
BCBUS3 tx_led
BCBUS2 rx_led
BCBUS1
BCBUS0 ft_txden
R1 470R
R2 470R
C4 100nF
C5 27pF
C8 27pF
Y1 6MHz
D_SRST YELLOW
USB mini B, 5-pin
VBUS
D-
D+
GND
SHLD
FB2
C6 100nF
C7 10uF
RA3 4 x 22R
R3 1k5R
R4 1k5R
U5 93C46
VCC
CS
ORG
D
SCL
Q
DU
GND
e_cs
e_data
e_clk
e_data_o
C9 100nF
D_USB
GREEN
R5 470R
pwren
C10 100nF
C11 100nF
C12 100nF
C13 100nF
C14 100nF
RA4 4 x 100kR
e_data_o
nsrst_oe
ntrst_oe
n_jtag_oe
U2A 74LVC2G125
U2B 74LVC2G125
U4A 74LVC2G125
U4B 74LVC2G125
U8 74LVC1G125
ntrst_oe
ft_ntrst
b_ntrst
nsrst_oe
ft_nsrst
b_nsrst
n_jtag_oe
ft_tck
b_tck
ft_tdi
b_tdi
ft_tms
b_tms
VCC
JVCC
RA1 4 x 22R
RA2 4 x 22R
b_ntrst
b_nsrst
b_tdo
b_tms
b_tdi
b_tck
ntrst
nsrst
tdo
tms
tdi
tck
JTAG
IDC20
RA5 4 x 100kR
U6 74LVC2T45
VCCA
VCCB
A1
A2
B1
B2
GND
DIR
ft_tdo
ft_nsrst_in
b_tdo
b_nsrst
R6 330R
D_JVCC
GREEN
U7 74LVC1G14
n_tvcc
C15 100nF
C16 100nF
C17 100nF
C18 100nF
C19 100nF
C20 100nF
J1
Header 3
+3.3V
JVCC
+5V
JTAG voltage select
no jumper - external JVCC
1-2 - 3.3V
2-3 - 5V
Title
JTAG-lock-pick, FT2232 & JTAG
Size
A4
Number
1/2
Revision
1.1
Date: 2010-01-31
Sheet of
File: D:\Elektronika\..\JTAG-lock-pick-main.SchDoc
Drawn By: Freddie Chopin



Title: JTAG-lock-pick, UART & RS-232		
Size: A4	Number: 2/2	Revision: 1.1
Date: 2010-01-31	Sheet of	
File: D:\Elektronika\..\JTAG-lock-pick-uart.SchDoc	Drawn By: Freddie Chopin	

9.2. Lista elementów (bill of materials) (wersja 1.1)

JTAG-lock-pick 1.1 - Bill Of Materials 31.01.2010

Designator	Description	Comment	Footprint	Model:Footprint	Value	Quantity
C1, C2, C7, C26	Polarized Tantalum Capacitor	Polarized capacitor	CAPMP3216X18M	Molded Capacitor, 2-Leads, Body 3,2x1,6mm, IPC Low Density	10uF	4
C3, C4, C6, C9, C10, C11, C12, C13, C14, C15, C16, C17, C18, C19, C20, C21, C24, C25, C28, C29, C30, C31	Capacitor	Capacitor	CAPC2012M	Chip Capacitor, Body 2.0x1.3mm, IPC Low Density	100nF	22
C5, C8	Capacitor	Capacitor	CAPC2012M	Chip Capacitor, Body 2.0x1.3mm, IPC Low Density	27pF	2
C22, C23, C27	Capacitor	Capacitor	CAPC2012M	Chip Capacitor, Body 2.0x1.3mm, IPC Low Density	470nF	3
D_JVCC, D_RS, D_UART, D_USB	Typical LED	GREEN	CAPC2012M	Chip Capacitor, Body 2.0x1.3mm, IPC Low Density		4
D_RX, D_SRST	Typical LED	YELLOW	CAPC2012M	Chip Capacitor, Body 2.0x1.3mm, IPC Low Density		2
D_TX	Typical LED	RED	CAPC2012M	Chip Capacitor, Body 2.0x1.3mm, IPC Low Density		1
FB1, FB2, FB3	Choke	Choke	RESC2012M	Chip Resistor, Body 2.0x1.3mm, IPC Low Density		3
J1, J3	Header, 3-Pin	Header 3	HDR1X3	Connector; Header; 3 Position		2
J2	Jumper Wire	Jumper	HDR1X2	Connector; Header; 2 Position		1
JTAG	Header, 10-Pin, Dual row	IDC20	IDC-20, angled	IDC header, angled, 20-pin		1
R1, R2, R5, R9, R10	Resistor	Resistor	RESC2012M	Chip Resistor, Body 2.0x1.3mm, IPC Low Density	470R	5
R3, R4	Resistor	Resistor	RESC2012M	Chip Resistor, Body 2.0x1.3mm, IPC Low Density	1k5R	2
R6, R7, R8	Resistor	Resistor	RESC2012M	Chip Resistor, Body 2.0x1.3mm, IPC Low Density	330R	3
RA1, RA2, RA3, RA6, RA7	Quad chip resistor array, 4D03	Resistor array	RESA3216X06M	Chip Resistor Array, 8-Leads, Body 3,2x1,6mm, IPC Low Density	4 x 22R	5
RA4, RA5, RA8	Quad chip resistor array, 4D03	Resistor array	RESA3216X06M	Chip Resistor Array, 8-Leads, Body 3,2x1,6mm, IPC Low Density	4 x 100kR	3
RS-232	Header, 5-Pin, Dual row	IDC10	IDC-10, angled	IDC header, angled, 10-pin		1
U1	Micropower 150 mA Low Noise Ultra Low-Dropout Regulator	LP2985-3.3	SOT-95P-284X119-5M	SOT23, 5-Leads, Body 3,0x3,0mm (max), Pitch 0,95mm, IPC Low Density		1
U2, U4, U11	Dual bus buffer gate with 3-state outputs	74LVC2G125	SOP-65P-400X130-8M	TSOP, 8-Leads, Body 3,2x2,9mm (max), Pitch 0,65mm, IPC Low Density		3
U3	Dual USB UART / FIFO I.C.	FT2232	TSQFP-50P-900X900X160-48M	TSQFP, 48-Leads, Body 9,0x9,0mm (max), Pitch 0,50mm, IPC Low Density		1
U5	1K (64 x 16 or 128 x 8) Serial Microwire EEPROM	93C46	SOIC127P600X175-8M	SOIC, 8-Leads, Body 5,0x4,0mm (max), Pitch 1,27mm, IPC Low Density		1
U6, U12	Dual-bit dual-supply bus transceiver with configurable voltage translation and 3-state outputs	74LVC2T45	SOP-65P-400X130-8M	TSOP, 8-Leads, Body 3,2x2,9mm (max), Pitch 0,65mm, IPC Low Density		2
U7, U10	Single Schmitt-trigger inverter	74LVC1G14	SOT-65P-212X110-5M	SOT23, 5-Leads, Body 2,3x2,3mm (max), Pitch 0,65mm, IPC Low Density		2
U8	Single bus buffer gate with 3-state output	74LVC1G125	SOT-65P-212X110-5M	SOT23, 5-Leads, Body 2,3x2,3mm (max), Pitch 0,65mm, IPC Low Density		1
U9	±15kV ESD-Protected, 1µA, 3.0V/5.5V, 250kbps, RS-232 Transceiver with AutoShutdown	MAX3243	SOP65P710X200-28M	SOP, 28-Leads, Body 10,3x5,4mm (max), Pitch 0,65mm, IPC Low Density		1
UART	Header, 7-Pin, Dual row	IDC14	IDC-14, angled	IDC header, angled, 14-pin		1
USB	USB 2.0, Right Angle, SMT, B Type, Receptacle, 5 Position, Black	mini B, 5-pin	USB mini-B SMD	Connector; USB2.0, Type B, SM; 5 Position; Right Angle		1
Y1	Crystal Oscillator	6MHz	HC-49S	Quartz Crystal SMD		1