

Atmel

生産する製造業者から: ArduinoからAVRへの間隙の橋渡し

AN-12077

事前必要条件

- ハート・ウェア事前必要条件
 - ATmega328P Xplained Mini基板
 - IO1 Xplained Pro拡張基板
 - Arduino Xplained Pro基板
 - マイクロUSBケーフル
- ソフトウェア事前必要条件
 - Atmel[®] Studio 6.2またはそれ以降版
 - Arduino IDE 1.6.0
 - Atmel Studio用Arduino拡張
 - 端末ウィントウ拡張
- 予想完了時間
 - 2時間

序説

この実践は豊富な使用者インターフェースとそれが提供する他の優れた開発ルールと共にAtmel Stu dioを使ってArduinoを開発する方法を実演します。

Arduinoは柔軟性、使い易いハードウェアとソフトウェアに基づく解放ソースの電子回路試作基盤です。 それは芸術家、設計者、趣味人、そして相互作用的ば物や環境を作成することに興味がある 誰かのために意図されています。Arduinoについての重要な事実はこれらの基板がAtmelマイク ロコントローラ系統に基づき、基本的なソフトウェアはAtmel開発ツールに基づきます。

'何故Aruduinoから切り替えるべきか?'に対する答えは、Arduino IDEが以下だからです。

- 経験豊富な書き手の能力に対するかなりの制限
- コンパイラ警告とデバッグ能力の不足
- (全ての命令文の後のSerial.println()は数ず)、高度なプロジェクトでの作業時に重荷
- Atmel Studioは統合されたArduino IEDを脱した使用者に対する素晴らしい選択
- それらのArduinoライブラリのいくつかは正に殆ど価値のない簡便性

従って何故両世界を出来るだけ利用しないのか?。Arduinoはデバッグ能力を持つC/C++の頂上の包装部(ラッパー)。どんなArduinoのスケッチやライブラリでも高度なプロジェクトであなた自身の独自コートと結合することが可能です。

Atmelは趣味人から開発者への移行の橋渡し、助けを提供するための"going pro"共同体についてのArduinoでの独特で特権的位置と責任を持ちます。

Atmel Xplained Mini系統はCとC++への容易な移行に対する完璧な"橋渡し"です。それはArd uinoに対して非常に類似した構造を持ち、Arduino基板のも飛渡の機能を提供します。IDEが 正しく構成設定された時にArduinoスケッチの走行が可能です。基板上のハートウェア デバッガ/書き 込み器も利用可能です。それは非常に一般的なAVR®マイクロコントローラ系統使い、相対的に安価です。



この訓練手引き書は4つの課題から成ります。

最初の課題ではATmega328P Xplained MiniをArduino IDEに接続L方法を理解します。

第2の課題ではAtmel StudioのArduino拡張を使ってAtmel StudioでArduinoスケッチをを作成します。

第3の課題では直接スケッチを取り入れるようにAtmel Studioを構成設定します。既存Arduinoス ケッチをAtmel Studioへ移行する方法を検討します。また、Atmel Studioで中断点(ブレークポイント) を挿入する方法とデベッグする方法を調べます。

第4の課題ではファイルを編集して周辺機能(ADC,I²C,SPI)を持つ簡単な応用を作成します。この応用はA/D変換器(ADC)を通し光感知器を読み、I²Cインターフェースを通して温度感知器を読み、そしてSPIインターフェースを通してSDカード内にデータを格納します。

•
11

1.	訓練	单位部構造·····	5
	1.1.	Atmel Studio拡張 (.vsix) ······	5
	1.2.	Atmel訓練実行物 (.exe) ···································	5
2.	課題1	: Arduino IDEへのATmega328P Xplained Mini接続方法 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	5
	2.1.	Arduino IDE	5
	2.2.	Atmega328P Xplain MiniでのmEDBGファームウェア格上げ(更新)・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	5
	2.3.	ATmega328Pでのブートローダヒューズ設定 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	7
	2.4.	ブートローダ書き込み ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	7
	2.5.	Arduino IDE構成設定 ······	8
	2.6.	プログラムのアップロード ・・・・・	10
3.	課題2	? : Atmel StudioでのArduinoスケッチ作成 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	1
	3.1.	拡張のダウンロード・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	11
	3.2.	スケッチ作成 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	11
4.	既存A	ArduinoスケッチのAtmel Studio IDEへの移行 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	4
	4.1.	プロジェクト作成 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	14
	4.2.	コンパイラ シンボル構成設定 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	16
	4.3.	コンパイラ ディレクトリ構成設定 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	16
	4.4.	Arduino依存ファイル追加 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	17
	4.5.	解決策構築 ••••••••••••••••••••••••••••••••••••	18
	4.6.	Atmega328P Xplined Mini基板差し込み ····································	18
	4.7.	<i>¯`N`yD</i> [`] ······ 1	19
5.	ATme	ga328P応用・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	23
	5.1.	コンパイラ構成設定 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	23
	5.2.	依存ファイル追加 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	24
		5.2.1. Wireライブラリに対して ······ 2	24
		5.2.2. SDライフ [*] ラリに対して・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	24
	5.3.	応用開発・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	25
	5.4.	<u>ハートウェア接続</u> ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	29
		5.4.1. 接続:IO1 Xplained Pro - Arduino Xplained Pro ···································	30
		5.4.2. 接続: A I mega328P Xplained Mini - Arduino Xplained Pro	31
	55	0.4.3. 按枕:USD7=7ル 広田の=゙バッガ	ン2 ン2
	0.0.)う)フ
~	2旦1開月	ヘ∶牀瓼41~ンジ9 ②元王は胜次束3) (
б. -	おい	4 = m	FO
7.	改訂和	复歴 • • • • • • • • • • • • • • • • • • •	f0

アイコン基本識別子

i	情報	特定の話題について文脈上の情報を提供
	助言	有用な助言と技法を強調
	実施事項	完了されるべき目標を強調
\checkmark	結果	課題段階の予測される結果を強調
	警告	重要な情報を表示
•	実行	必要時に目的対象の実行されるべき活動を強調



1. 訓練単位部構造

この訓練素材は以下として各種Atmel配給物を通して取得することができます。

- 通常、Atmel展示室ウェブ サイト(http://gallery.atmel.com/)で、またはAtmel Studio拡張管理部を使って得られるAtmel Studio拡張 (.vsixファイル)
- 通常、Atmel訓練作業中に提供されるAtmel訓練実行物(.exeファイル)

配布形式に依存して、この訓練素材によって必要とされる各種資料(実践訓練資料、データシート、応用記述、ツール)は種々の場所で得られます。

1.1. Atmel Studio拡張 (.vsix)

一旦この拡張がインストールされると、Atmel Studioで"New Example Project from ASF…"を使って種々のプロジェクトを開いて作成することができます。

- う この拡張でインストールされたプロジェクトは一般的に"Atmlel Training⇒Ateml Cort. Extension Name"下で得られます。

拡張に依存して利用可能な各種プロジェクトがあります。

- Hands-on Documentation (実践訓練資料): 必要とされる資源としての資料を含みます。
- Hands-on Assignment (実践訓練課題) :開始に必要とされるかもしれない初期プロジェクトを含みます。
- Hands-on Solution (実践訓練解決策) : この実践訓練に対する解決策である最終的な応用を含みます。



1.2. Atmel訓練実行物 (.exe)

実行物がインストールされた場所に依存して、2つの主なフォルダから成る以下の基本構造を見つけるでしょう。

- AN-12077_Hands-on : 開始と解決策に必要とされるかもしれない初期プロジェクトを含みます。
- Resources : 必要とされる資源(データシート、ソフトウェア、ツールなど)を含みます。



特定の位置が指定されない限り、以降の頁でいくつかの資源に対して参照が行われる度に、使用者はこのResources(資源)プロジェクトフォルダを参照しなければなりません。

2. 課題1 : Arduino IDEへのATmega328P Xplained Mini接続方法

2.1. Arduino IDE

www.arduino.ccからArduino IDEをダウンロードしてください。

2.2. ATmega328P Xplained MiniでのmEDBGファームウェア格上げ(更新)

- 1. Atmel作業空間(http://spaces.atmel.com/gf/project/avr_xp_mini/frs/)へ行ってください。"medbgdebugger"一括からmedbg_f w.zipを選んでダウンロードしてください。
- 2. Atmel Studio インストール フルダ (例えば、C:¥Program Files (x86)¥Atmel¥Atmel Studio 6.2¥tools¥mEDBG)にmedbg_fw.zipのZIP一括 を上書きしてください。
- 3. Atmel Studioを開始してください。
- 4. ATmega328P Xplained Miniをコンピュータに接続してください。
- 5. Atmel Studioに於いて、Tools⇒Device programming (alt.:Ctrl+Shift+P)を選んでください。
- 6. デバイスプログラミング ウィンドウで、ToolにmEDBGを設定して"Apply"をクリックしてさい。

mEDBG (ATML	.2323040200	0011546) - C	evice Progra	amming							? 💌
Tool	Device		Interface		D	evice signature		Target Volt	age		
mEDBG 🔻	ATmega32	28P 🔻	ISP •	Apply	0	x1E950F	Read	5,0 V	Read		
Interface sett	tings	mEDBG									
Tool informa	ation	Debug ho	ost		127.0	0.0.1					
Device inforr	mation	Debug po	ort		5108	6					
Oscillator Ca	libration	Serial nur	nber		ATMI	L2323040200011	.546				
Memories		Connectio	on		com.a	atmel.avrdbg.cor	nnection.cr	msis-dap			
Fuses		Firmware Version			1.6						
Lock bits		Hardware Version			0						
Production fi	ile	External Li	nk: Information								Copy to clipboard
Getting	tool info	.ОК	_	_		_	_	_			Close

情報 次にAtumel Studioがファームウェアを格上げ(更新)を望むかを尋ねるでしょう。

Firmware Upgrade		8					
EDBG firmware is upo	lated						
	On Tool	On Disk					
Firmware Version	1.0	1.6					
EDBG firmware successfully upgraded							
EDBG firmware succe	sstully upgraded						

7. Upgradeを選んでください。



ft報 Atmel Studioのプログラシッグ/デバッグによる認証をできなくする、いくつかのATmega328P Xplained Mini基板での通番のパグがあります。あなたの基板が通番内に未知の文字を持つ場合、"Releases"フォルダ内に修正があります。あなたの基板の通番を見るには、Atmel Studioを開始してTools⇒Device programmingでツールを選択してください。これは通番と共にmEDBGを一覧にし、いくつかの文字が黒背景で"?"を持つ場合、通番修正(serial number fix)から一括をダヴンロートしてHow_to_change_serial_Number.pdfからの指示に従ってください。

Atmel

2.3. ATmega328Pでのブートローダヒューズ設定

1. 次に'Device Programming'ウィンドウで、'Fuses'を選択してください。

- 2. 下で示されるようにEXTENDED,HIGH,LOWの値を変更してProgramをクリックしてください。
 - EXRENDED = \$FF
 - HIGH =\$D8
 - LOW = E0

☑2−1. Device Programming : Fuses									
mEDBG (ATML2323020200000190) - Device Programming									
Tool Device Interface				Device signatur	Target Vo	ltage			
mEDBG ATmega3	28P 🔻 ISI	P •	Apply	0x1E950F	Read	5,0 V	Read		
Interface settings	Fuse Name		Va	lue					*
Tool information	SPIEN	V							
Device information	WDTON								
Oscillator Calibration	V EESAVE								=
Memories	Ø BOOTSZ	2048	W_3800	•					
Fuses	BOOTRST								
Lock bits	CKDIV8								-
Production file	Fuse Register	Value							
	EXTENDED	0xFF							
	HIGH	0xD8							
	LOW	0xE0							
	Auto read							Сору	to clipboard
	Verify after	program	ming			Pro	gram	Verify	Read
Verify registersOK Verify register LOWOK Verify register LOWOK Verify registersOK									
								l	Close

2.4. ブートローダ書き込み



ブートローダ hexファイルはC:¥Program Files (x86)¥Arduino¥hardware¥arduino¥avr¥bootloaders¥atmega¥*.hexのArduino IDEフォルダに置かれます。 ブートローダは表2-1.で一覧されるように基板構成設定に従って選択することができます。

表2-1.7-トローダ							
Xplained Mini	フ゛ートロータ゛						
ATmega328P/5V/16MHz	ATmegaBOOT_168_atmega328.hex						
ATmega168P/5V/16MHz	ATmegaBOOT_168_ng.hex						
ATmega168P/3.3V/8MHz	ATmegaBOOT_168_pro_8MHz.hex						

- 1. 'Tools⇒Device Programming'でのウィンドウで、'Memories'タブを選んでください。
- 2. C:¥Program Files (x86)¥Arduino¥hardware¥arduino¥avr¥bootloaders¥atmega¥ ATmegaBOOT_168_atmega328.hexを閲覧してく ださい。
- 3. Programをクリックしてください



]2−2. Device Programming : Memories								
mEDBG (ATML2323040200	0011546) - Device Programming							
Tool Device mEDBG ATmega32	Interface Device signature Target Voltage 28P ISP Apply Device signature Target Voltage Read Comparison Read Comparison Comparison Compariso							
Interface settings Tool information Device information Oscillator Calibration	Device Erase Chip Erase now Flash (32 KB) C:\Program Files (x86)\Arduino\hardware\arduino\avr\bootloaders\atmega\ATmegaBOOT_168_atmega328.hex							
Memories Fuses Lock bits	Image: Second							
Production file	Verify EEPROM after programming Program Verify Read							
 OK 	Close							

2.5. Arduino IDE構成設定

- 1. Windows®のスタート メニューまたはArduino IDEがインストールされたフォルタからArduino IDEを開始してください。
- 2. メニューからのTools⇒PortからでmEDBG用の正しいCOMポートを選択してください(mEDBGに使われるCOMポートを確認する方法を 次の助言でご覧ください)。

図2-3. Ardduino IDE	E : (Serial) Port			
💿 sketch_feb25c /	Arduino 1.6.0			
File Edit Sketch To	ols Help			
sketch_feb25c void setup() { // put your se	Auto Format Archive Sketch Fix Encoding & Reload Serial Monitor	Ctrl+T Ctrl+Shift+M		
)	Board Processor		•	
// put your ma	Port			COM3
}	Programmer Burn Bootloader		 ✓ 	COM27





3. 基板(Board)は表2-2.に従って選択することができます。

表2-2. Arduino IDE : Board						
Xplained Mini	Arduino IDEで選択する基板					
ATmega328P Xplained Mini	Arduino Nano : ATmega328					
ATmega168PB Xplained Mini	Arduino Nano : ATmega168					

ここで、選択: Tools⇒Boards⇒Arduino Nano 選択: Tools⇒Processor⇒ATmega328

sketch_feb25c	Arduino 1.6.0		sketch_feb25c Arduino 1.6.0						
File Edit Sketch To	pols Help		File Edit Sketch	pols Help					
sketch_feb25c	Auto Format Ctrl+T Archive Sketch Fix Encoding & Reload Serial Monitor Ctrl+Shift+M		<pre>void loop() (// put your se) void loop() (// put your me)</pre>	Auto Format Ctrl+T Archive Sketch Fix Encoding & Reload Serial Monitor Ctrl+Shift+N					
3	Board	Arduino AVR Boards		Board					
	Processor	Arduino Yún		Processor		ATmega328			
// put your ma	Port	Arduino Uno		Port	1	ATmega168			
3	Programmer Burn Bootloader	Arduino Duemilanove or Diecimila Arduino Nano Arduino Mena or Mena 2560		Programmer Burn Bootloader					
		Arduino Mega of Mega 2000 Arduino Leonardo Arduino Micro Arduino Esplora Arduino Mini Arduino Ethernet							

Atmel

9

2.6. フ[°]ロク^{*}ラムのアッフ[°]ロート^{*}

🖌 結果 ATmega328P Xplained Mini基板はArduino IDEへ接続され、LED点滅が始まるでしょう。

1. Arduino IDEで、Files⇒Examples⇒01.Basics⇒Blinkを選択してください。

圛2-	-5. Ardduino IDE : Exampl	es-Blink				
00	sketch_feb25c Arduino 1.6.0)				
File	Edit Sketch Tools Help					
	New	Ctrl+N				
	Open	Ctrl+O				
	Sketchbook)				
	Examples	,		01.Basics	1	AnalogReadSerial
	Close	Ctrl+W		02.Digital	1_	BareMinimum
	Save	Ctrl+S		03.Analog	1	Blink
	Save As	Ctrl+Shift+S		04.Communication	1	DigitalReadSerial
	Upload	Ctrl+U		05.Control	1	Fade
	Upload Using Programmer	Ctrl+Shift+U		06.Sensors	1	ReadAnalogVoltage
	Page Setup	Ctrl+Shift+P		07.Display	*	
	Print	Ctrl+P		08.Strings 09.USB	*	
	Preferences	Ctrl+Comma		10.StarterKit	•	
	Quit	Ctrl+Q		ArduinoISP		
				Audio	•	
				Bridge	•	
				EEPROM	*	
				Esplora	•	
				Ethernet	•	
				Firmata		

2. Arduino IDEで"Upload" 釦をクリックすることによってスケッチをアップロードしてください。



✓ 結果

ウィントウの下部で'Compiling Sketch(スケッチをコンパイル中)'、'Uploading(アップロート`中)'、'Done uploading(アップロート`終 ・ 了)'のメッセーシ`が出現されるでしょう。

- 3. File⇒Save As...を選んで'Blink'スケッチを保存してください。スケッチを保存するのにどのパスをも選ぶことができます。
- 4. ATmega328P Xplained Mini基板でLED点滅を観察してください。

これで成功裏にArduino IDEへ接続されたATmega328P Xplained Mini基板があります。

3. 課題2 : Atmel StudioでのArduinoスケッチ作成

情報 Atmel StudioのArduino拡張はそれが提供する豊富な使用者インターフェースや専門的な機能を持つAtmel Studio内で書い てコンハプイルし、そしてどのArduinoへもアップロードすることをArduinoのスケッチに許します。

3.1. 拡張のダウンロード

- 1. Atmel Studioを開いてください。
- 2. Tools⇒Extension Managerを選んでください。 拡張管理部 (Extension Manager)ウィントウが開き、 既定によってインストールされている 拡張を示します。
- 3. "Available Download"任意選択をクリックしてください。

⊠3-1. Atmel Studio : Extension Manager Extension Manager ? × Installed Extensions Sort by: Highest Ranked . Q Available Downloads Free Arduino IDE for Atmel Studio -All Compatible with all Arduino versions including 1.6. Download Arduino for Atmel Studio is a simple and fully comp. Updates (4) Visual LUFA Library 21 Micro LUFA, the Lightweight USB Framework for AVRs. Created by: Arduino Team @ Visual Micr Version: 1502.14 Free XFeatures - Beta Downloads: 113857 Nice to have features in atmel studio. Rating: ***** Project Enhancements: More Information Getting Started **Terminal for Atmel Studio** Free 0 An extension for a terminal window inside Atmel Reviews Studio providing serial port communication. ***** Code VisionAVR C Compiler Evaluation HAOCHEN WANG Free 2/16/2015 Version: 1412.10 Code size limited Evaluation version of the CodeVisionAVR C Compiler and CodeWizard for the ... ***** guardia guardia FreeRTOS+Trace Free 2/12/2015 Version: 1412.10 Ð Gain an unprecedented insight into the runtime world ***** of your FreeRTOS system using this powerful toolbox.. umar abdul 2/6/2015 Version: 1412.10 Free Naggy ***** Compiler diagnostics and preprocessor lowlighting in Robert Kloiber Atmel Studio for C and C++ projects. 2/5/2015 Version: 1412.10 ***** **Atmel Data Visualizer** Juan Pablo Cordova Atmel Data Visualizer is a program used for 2/5/2015 Version: 1412.10 just excellent !! 1 2 3 4 1 Close

4. "Arduino IDE for Atmel Studio"を選んで"Download"アイコンをクリックしてください。

5. "Sign in to Extension Manager Dashboard"ウィントウが開き、サインイン(sign-in)/登録(register)を尋ねます。

- 情報 既に登録していないなら、どうぞ登録してください。その後に"Sign in to Extension Manager Dashboard"ウィンドウを閉 じてください。Atmelは提供されるEメールIDのために、あなたへ確認Eメールを送ります。あなたのEメールIDを確認するた めにリンクをクリックしてください。"Sign in to Extension Manager Dashboard"で手順2.~4.を繰り返してください。Eメール IDと登録中に提供されるパスワードを使ってサインインしてください。
- 6. "Arduino IDE for Atmel Studio" 拡張がダウンロート され、その後にそれをインストールしてください。

3.2. スケッチ作成

1. Atmel Studioを開いてください。

🔛 実施事項

Atmel StudioでArduino 1.6が一覧にされていることを確認し、それを選んでください。





情報 それが一覧にされてなければ、〈Configuration Manager〉をクリックしてください。Arduino 1.6を選んでください。下図で 示されるようにArduinoインストール フォルダのパスを追加してください。'OK'を選んでください。

Config	ure Visual Micro						
Win	Please specify a micro-controller Ide location						
	Visual Micro needs to know where, on your computer, application(s) such as the Arduino.exe are located.						
	If an application is not already installed then please download it using the 'Download/Install' button. Support for some platforms is still under development and we value feed back in our forum						
	Arduino 1.6 👻						
	Enter the ide folder location (example: c:\arduino)						
	c:\Program Files (x86)\Arduino						
	Optionally enter a sketchbook location (also affects location of custom libraries/hardware!)						
	C:\Arduino						
	Download/Install Ide OK Cancel						
WA H	elp and information						
Н	ow to test a new installation						
	atting started						

2. File⇒New⇒Sketch Projectを選んでください。

3. 下図で示されるようにスケッチプロジェクト用の名前を入力して'OK'を選んでください。

Create a new arduino sketch project?	×
Enter a name for the new sketch	ОК
	Cancel
led_toggle	

↑ 情報 led_toggle.inoスケッチは図3-2.で示されるように既定位置としてc:Arduinoに作成されるでしょう。

4. LEDを交互ON/OFFすようにコードを編集してください。

```
int led = 13;
void setup()
{
    /* 準備コート をここに追加してください。*/
    pinMode(led, OUTPUT);
}
void loop()
{
    /* 主プ<sup>°</sup>ログラムコート をここに追加してください。*/
    digitalWrite(led, HIGH);
    delay(500);
    digitalWrite(led, LOW);
    delay(500);
}
```

5. 基板をArduino Nano w/ATmega328として選んでください。

🗫 led_toggle - AtmelStudio						
File Edit View	VAssistX ASF Project Build Debug Tools W	indo				
🛅 🕶 🕮 🖘 🞽	। 🛅 • 🕮 📨 📂 🛃 🍠 🔉 🧤 🔁 🗠 🕫 - 🔍 - 🕮 - 🖳 🔍					
i 😨 ሯ 🞇 🍋 皆	i 🗒 🖾 🖓 🍋 🔓 🥵 🕵 🚅 🚽 i 💵 💷 💠 🗉 🕨 🕼 🗐 🗊					
Arduino 1.6 🝷	Arduino Nanc 🔻 🧣 🚽 🛛 COM3 🛛 🗣 🌉 💷 🕔					
led_toggle.ino ×	Arduino Yún Arduino Uno					
→ loop	Arduino Duemilanove or Diecimila w/ ATmega328					
int led =	Arduino Duemilanove or Diecimila w/ ATmega168					
<pre> void setu { /* add pinMod } </pre>	Arduino Nano w/ ATmega328 Arduino Nano w/ ATmega168 Arduino Mega w/ ATmega2560 (Mega 2560) Arduino Mega w/ ATmega1280 Arduino Mega ADK Arduino Leonardo Arduino Micro	H				
<pre> Vold loop { /* add digita delay digita delay digita </pre>	Arduino Esplora Arduino Mini w/ ATmega328 Arduino Mini w/ ATmega168 Arduino Ethernet Arduino Fio Arduino BT w/ ATmega328 Arduino BT w/ ATmega328					

6. mEDBGのCOMポート番号を選んでください。

秦 led_toggle - AtmelStudio
File Edit View VAssistX ASF Project Build Debug Tools Window Help
🗄 🐂 📲 🕮 - 📂 🛃 🌒 🐇 💺 隆 🖉 - 🔍 - 🕮 - 🖳 🔛 🍕 🌍 Debug
। 🐑 🗁 🐼 🍋 🎦 🚰 💁 🔬 🗳 🚽 🕅 💷 🌩 🗉 🕨 🍛 🖘 🍹 🗊 🖅 Hex 🚳 🕶
🕴 Arduino 1.6 🔹 Arduino Nanc 🔹 🎖 📮 🕴 COM3 🔹 🔜 💷 📮
COM3 - Intel(R) Active Management Technology - SOL
COM27 - mEDBG Virtual COM Port

▶ アイコンをクリックすることによってプログラムをダウンロートしてください。

結果 ATmega328P Xplained Mini基板上のLEDが交互ON/OFFを始めるでしょう。

4. 既存ArduinoスケッチのAtmel Studio IDEへの移行

4.1. プロジェクト作成

- 1. Atmel Studioを開き、"GCC C++ Executable Project"で新規プロジェクトを作成し、妥当な名前を与え、デバイスとしてATmega328P を選んでください。
- 2. プロジェクトが作成されたフォルダの場所を記録することを確実にしてください。



3. 結果のSolution Explorerで、下図で示されるように右クリックしてプロジェクトCPPファイル(ここではATmega328P_1.cpp)を削除してください。

File Edit View VAssistX ASF Project Build Debug Tools Window Help		
🔁 + 🕮 🖃 - 🥁 😹 🐇 🛵 🐮 🔊 - C* + 💭 - 🖳 🐼 🔍 🌹 🕨 Debug 🔹 🧖 main 🔹 🔹 🖓 🚏 🐼 🔮	20 · · · · · · · · · · · · · · · · · · ·	-
🗖 🗃 🖉 🕤 🖓 😗 🗸 🔔 🚛 👘 🖬 🚽 🖓 🚛 🖓 🧐 🕼 🐨 🖬 Hex 🍓 📲 🖓 🖬 🖉 🖉 👘 👘 👘 👘 👘 👘 👘 👘 👘 👘	No Tool 🖕	
Arduine 10x + • • • • • • • • • • •		
ATmensi228D Jans V ATmensi228D Valsiend Mini, 1546 ASE Minard	- Solution Explorer	* 1 ×
Annegaszor_szepi A Annegaszor Aplained Mini-1540 Ast Mizard	E Pol o	- 1.0
ATmega328P 1, cpp		
*	ATmega328P_1 (1 project)	
* Created: 25.02.2015 12:15:48	Dependencies	
* Author: rupali.honrao	🖼 Output Files	
	> El Libraries	
and the second se	Almega328P_1.cpp	
#include <avr 10.h=""></avr>	L Oper	1
Bint main(void)	Oper	i With
(😂 Oper	s File Location
while(1)	Сору	Full Path
{ //TODO:: Please write your application code //TODO:: Please write your application code	S View	Code Ctrl+Alt+0
3	E	pile Ctrl+F7
	👗 Cut	Ctrl+X
	Copy	Ctrl+C
	Rema	ove Del
	Rena	me r2
	ASE 1	Wizard
	SI Com	pare.
	•7 Beve	rt to original ASE file
	P. Pere	and as
	TER Prop	ziues

4. Windowsのエクスプローラで、

- a. あなたのスケッチ(先に保存したArduino点滅スケッチ/Blink.ino)を複写し、(例えAtmel Sudioのプロジェクトから取り去られたとしても、 未だエクスプローラにある)ATmega328P_1.cppのあるAtmel Studioのプロジェクト副フォルタ にそれを置いてください。
- b. あなたのスケッチ拡張を.cppに(Blink.ino⇒Blink.cpp)改名してください。下図をご覧ください。

Computer ► Local Disk (C:) ► workspace ► ATmega328P_1 ► ATmega328P_1 ► Organize ▼ Include in library ▼ Share with ▼ New folder						
눆 Favorites	Name	Date modified				
💻 Desktop	👩 Debug	25.02.2015 12:15				
🔈 Downloads	街 ATmega328P_1.cpp	25.02.2015 12:15				
🐉 Recent Places	🧑 ATmega328P_1.cppproj	25.02.2015 12:15				
	👸 Blink.cpp	25.02.2015 10:48				
🔲 Desktop						
🥞 Libraries						
🔊 Documents						
👩 Git						

5. Atmel StudioでProject⇒Show All Filesを選んでください。

🏶 ATmega_328P_1 - AtmelStudio					
File Edit View VAssistX ASF	Project Build Debug Tools Window Help				
🔁 • 🔁 🖂 • 🚰 📕 🍠 🐇 🛛	Add New Sketch Item	1			
: 🔁 📴 🐺 🍋 🕆 🤆 🛛	Add/Import Sketch Library	+			
Arduino 1.6 • Arduino Nanc •	K - 🔚 ASF Wizard				
	Show All Files				
	Set as StartUp Project				
	ATmega_328P_1 Properties Alt+F7	Tmega_328P_1 Properties Alt+F7			
	Show Sketch Core and Library Sources				

6. Solution Explorerウィンドウで、Blink.cppファイルを右クリックし右図で示されるよう に"Include In Project"を選んでください。



4.2. コンパイラ シンホル構成設定

1. Solution Explorerでプロジェクトを右クリックし、Propertiesを選ぶか、またはファイルメニュー(Alt+F7)でProject⇒プロジェクト名 Properties… へ行ってください。

2. Toolchain⇒AVR/GNU C Compiler⇒Symbols下に以下を追加してください。

F_CPU=16000000L USB_VID=null USB_PID=null ARDUINO=160

- 情報 これらはArduino特有シンボルです。
- 3. Toolchain⇒AVR/GNU C++ Compiler⇒Symbolsに対して繰り返してください。

🗭 ATmega328P_1 - AtmelStudio					
File Edit View VAssistX ASF Project Build Debug Tools Window Help					
🔁 • 🖞 🗁 • 😰 🚽 🖉 🐇 🖧 🖄 🔊 • 🔍 • 💭 • 🖏 🔚 🍳 🌹 🚸 🚧 Debug 🔹 🞯 main 🔹 💀 😤 🖉 🖉 🖉 🖉	.[注注]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]				
👦 🍃 🃓 🐂 😚 🥵 🚓 👔 🕴 💷 🗘 🚛 🗘 💷 🖡 💷 🐨 Hex 📑 📲 👘 🐨 🖬 🖉 👘 🗮 👘 👘 👘	Tool =				
Arching 10x + Y = COM3 + S = -					
ATmega328P_1* ×	Solution Explorer				
	I 🗳 🔅				
Build Configuration: Active (Debug) Platform: Active (AVR)	Solution 'ATmega328P_1' (1 project)				
Build Events	ATmega328P_1				
Toolchain* Configuration Manager	Qutput Files				
Device	▶ 🖼 Libraries				
Tool ▲ ▲ AWR/GNU Common AVR/GNU C Compiler → Symbols	🔛 🔛 🔛 🔛				
Advenced CutputFiles Defined symbols (-)					
Advanced AVE/GNU C Compiler					
General DLDOG					
USE_VID=null					
USB_PID=null APDI IND_160					
Optimization Children Control					
Miscellaneous Undefined symbols (-U)					
A @ AVR/SNU C++ Compiler General					
☐ Preprocessor					
Symbols Symbols					
2 Directories					
Cebugging					
Warnings					
Miscellaneous					
General					
Coptimization					

4.3. コンパイラ ディレクトリ構成設定

- 1. Toolchain⇒AVR/GNU C Compiler⇒Directories下に以下を追加してください。
- 2. C:\Program Files (x86)\Arduino\hardware\arduino\avr\cores\arduino
- 3. C:¥Program Files (x86)¥Arduino¥hardware¥arduino¥avr¥variants¥standard
- 4. C:\Program Files (x86)\Arduino\Hardware\Hardware\Hardwino\Hardware\Hardware\Hardwino\Hardware
- 5. Toolchain⇒AVR/GNU C++ Compiler⇒Directoriesに対して繰り返してください。
 - 警告 これらのパス追加中に"Relative Path"の選択解除を確実にしてください。



4.4. Arduino依存ファイル追加

1. プロジェクトを右クリックして"Add⇒Existing Item…"に行ってください。

- 2. 開いたダイアログ枠で閲覧部のパスC:¥Program Files (x86)¥Arduino¥hardware¥arduino¥avr¥cores¥arduinoを狙ってください。
 - a. "File name" 枠で*.c*<Enter>を入力してください。
 - b. 全ファイルを多選択して"Add as Link"を選んでください。

anize • New fold	der				•	0	
Atmel Studio 6.2	Name	Date modified	Туре	Size		*	- Solution Explore
6.2	👸 abi.cpp	08.02.2015 23:39	C++ Source File	2 KB			D. La
	👸 CDC.cpp	08.02.2015 23:39	C++ Source File	6 KB			Colution 'A'
Favorites	💕 HardwareSerial.cpp	08.02.2015 23:39	C++ Source File	8 KB			A ATmega
Desktop	街 HardwareSerial0.cpp	08.02.2015 23:39	C++ Source File	3 KB			🔤 Depe
Downloads	HardwareSerial1.cpp	08.02.2015 23:39	C++ Source File	3 KB			🔤 Outpu
Recent Places	HardwareSerial2.cpp	08.02.2015 23:39	C++ Source File	2 KB			Librar
	街 HardwareSerial3.cpp	08.02.2015 23:39	C++ Source File	2 KB			
Desktop	街 HID.cpp	08.02.2015 23:39	C++ Source File	14 KB			
Libraries	hooks.c	08.02.2015 23:39	C Source File	2 KB			
Documents	IPAddress.cpp	08.02.2015 23:39	C++ Source File	2 KB			
a Git	💕 main.cpp	08.02.2015 23:39	C++ Source File	2 KB			
🔬 Music	💕 new.cpp	08.02.2015 23:39	C++ Source File	2 KB		=	
Pictures	Ø Print.cpp	08.02.2015 23:39	C++ Source File	6 KB			
a Subversion	👸 Stream.cpp	08.02.2015 23:39	C++ Source File	8 KB			
Videos	街 Tone.cpp	08.02.2015 23:39	C++ Source File	15 KB			
Honrao, Rupali	街 USBCore.cpp	08.02.2015 23:39	C++ Source File	14 KB			
Computer	 WInterrupts.c 	08.02.2015 23:39	C Source File	9 KB			
Network	👩 wiring.c	08.02.2015 23:39	C Source File	10 KB			
Control Panel	øiring_analog.c	08.02.2015 23:39	C Source File	8 KB			
Recycle Bin	i wiring_digital.c	08.02.2015 23:39	C Source File	5 KB			
the state	øiring_pulse.c	08.02.2015 23:39	C Source File	4 KB			
	øiring_shift.c	08.02.2015 23:39	C Source File	2 KB			
	💕 WMath.cpp	08.02.2015 23:39	C++ Source File	2 KB			
	💕 WString.cpp	08.02.2015 23:39	C++ Source File	17 KB		-	
Filo pa	me: "abi con" "CDC con" "HardwareSeri	al cop" "HardwareSerialO.cop" "	HardwareSerial1.cop" '	HardwareSorial'	All Filos (* *)	-	

Atmel

4.5. 解決策構築

1. Blink.cppの先頭に# include "arduino.h"を追加してください。



- 情報 今や全てのAtduino依存物が追加され、プロジェクトをコンパイルして目的対象内に読み込むことができます。
- 2. ファイル メニューで、Build⇒Build Solutionを選んでください。この構築は誤りなしで成功裏に終わるべきです。



4.6. ATmega328P Xplained Mini基板差し込み

- 1. Solution Explorerでプロジェクトを右クリックしてファイル メニュー(Alt+F7)でPropertiesを選ぶか、またはProject⇒プロジェクト名 Properties… へ行ってください。
- 2. Tool下で、InterfaceとしてmEDBGとdebugWIREを選んでください。

Image 328P_1 - AtmelStudio File Edit View VAssistX ASF Project Build Debug Tools Window Help Image 328P_1 - AtmelStudio Image 328P_1 - Atmelstu	월 1월
Blinkcpp ATmega328P_1* × Build Build Build Events Toolchain Device Tool Advanced Programming settings Erase only program area Preserve EEPROM Debug settings Ø Cache all flash memory except	Solution Explorer Solution ATmega328P_1' (1 project) Solution 'ATmega328P_1 Dependencies Solution 'ATmega328P_1 Addressolution 'ATmega328P_1 Solution 'Atmega328P_1 Sol

情報 いくつかの箇所でシステムはディッガの更新を望むかもしれません。それを更新しましょう。

- 3. Toolchain⇒AVR/GNU C Compiler⇒Optimization下で最適化レヘル'のptimize for size(-Os)'を選んでください。
- 4. Toolchain⇒AVR/GNU C++ Compiler⇒Optimization下で最適化レヘル'Optimize for size(-Os)'を選んでください。

Build Build Events	Configuration: Active (Debug)	Platform: Active (AVR)
Toolchain	Configuration Manager	
Device Tool	AVR/GNU Common	AVR/GNU C++ Compiler 🌩 Optimization
Advanced	 General OutputFiles General Preprocessor Symbols Directories Optimization Debugging Warnings Miscellaneous AVR/GNU C++ Compiler General Preprocessor Symbols Directories Optimization 	Optimization Level: Optimize for size (-Os) Other optimization flags: None (-O0) Optimize for size (-O1) Optimize more (-O2) Optimize more (-O2) Optimize most (-O3) Ø Prepare data for garbage Optimize for size (-Os) Ø Pack Structure members together (-fpack-struct) Ø Allocate only as many bytes needed by enum types (-fshort-enums) Use rjmp/rcall (limited range) on >8K devices (-mshort-calls) Image: Construct of the struct of the
	Debugging Warnings Miscellaneous	

4.7. デバック゛

1. Debugを選び、'Start Debugging and Break'をクリックしてください。

ATmega328P_1 - AtmelStudio					
File Edit View VAssistX ASF Project Build	Deb	ug Tools Window Help			
i 🐜 = 🕮 💷 = 😂 🛃 🛃 🕷 🐁 🛍 🖄 🖛 🤊 = 🤆	Windows •				
i 🔁 📴 🐺 🍋 🔓 🥵 🚱 🖕 i 📖 🗉 📔	D II	Start Debugging and Break	Alt+F5		
Arduino 1.0.x 🔹 🔹 😵 🚽 COM3		Stop Debugging	Ctrl+Shift+F5		
Blink cnn X ATmega328P 1*	-≣⊅	Start Without Debugging	Ctrl+Alt+F5		
		Disable debugWIRE and Close			
pin the on-board LED is connected		Continue	F5		
the documentation at <u>http://ardui</u>	÷	Execute Stimulifile			
This example code is in the publi	÷	Set Stimulifile			
This example code is in the publi	5	Restart			
modified 8 May 2014	63	QuickWatch	Shift+F9		
by Scott Fitzgerald	۶I	Step Into	F11		
L /	ÇI	Step Over	F10		
<pre># include "arduino.h"</pre>	Ē	Step Out	Shift+F11		
<pre>// the setup function runs once whe</pre>	*3	Run To Cursor	Ctrl+F10		
// initialize digital pin 13 as a	Î	Reset	Shift+F5		
<pre>pinMode(13, OUTPUT);</pre>		Percepio Trace	•		
_ }		Toggle Breakpoint	F9		
<pre>// the loop function runs over and</pre>		New Breakpoint	•		
□void loop() {	2	Delete All Breakpoints	Ctrl+Shift+F9		
digitalWrite(13, HIGH); // turr delav(1000); // wait		Clear All DataTips			
<pre>digitalWrite(13, LOW); // turr</pre>		Export DataTips			
delay(1000); // wait		Import DataTips			
Γ1		Options and Settings			

警告 DWENヒューズが許可されていない場合に異常メッセージが表示されます。下で示されるように'Yes'をクリックしてヒューズを 設定するのにAtmel StudioはISPを使います。



結果 デバッガが開始され、mainで中断されます。今やデバッグを開始する準備が整いました。



情報 Debugメニューでは種々のデバッグ任意選択が利用可能です。

File Edit View VAssistX ASF Project Build	Deb	oug Tools Window Help		
🗛 📲 🕮 🖬 🖉 📓 🖉 🕹 🖽 🖄 🖓		Windows	٠	SPLQUARTER_SPEED -
i 🖸 😅 🖓 🍋 🎖 铬 🗛 🔬 🗳 🚽 💷 🔳	911	Start Debugging and Break	Alt+F5	
🕴 Arduino 1.0.x 🔹 Arduino Nanc 📲 🚽 COM28		Stop Debugging	Ctrl+Shift+F5	
Disassembly main.cpp x Blink.cpp Atmo	Þ	Start Without Debugging Disable debugWIRE and Close	Ctrl+Alt+F5	
#include <arduino.h></arduino.h>	* 4	Continue Execute Stimulifile	F5	
<pre> int main(void) { init(): </pre>	₩ 8	Set Stimulifile Restart		
	-11	Break All	Ctri+F5	
#if defined(USBCON)	61	QuickWatch	Shift+F9	
USBDevice.attach(); #endif	®I	Step Into	F11	
	(J)	Step Over	F10	
<pre>setup();</pre>	2	Step Out	Shift+F11	
for (::) {	•1	Run To Cursor	Ctrl+F10	
loop();	Ŧ	Reset	Shift+F5	
if (serialEventRun) serial		Percepio Trace	•	
		Toggle Breakpoint	F9	
return 0;		New Breakpoint	•	
_}	2	Delete All Breakpoints	Ctrl+Shift+F9	
		Clear All DataTips		
		Export DataTips		
		Import DataTips		
		Options and Settings		

2. ('Solution Explorer'のBlink.cppファイルで)中断点(ブレーク ポイント)の挿入を望む場所であるソース コード内の行へ行き、右クリックしてBr eakpoint⇒Insert Breakpointを選んでください。





結果 中断点(ブレーク ポイント)が挿入されます。



3. "Continue"をクリックすることによって中断点へ走行してください。必要な度毎に実行を一時停止と継続をすることができます。

ile	Edit Viev	v VAssistX	ASF Project	Build De	ebug T	ools Window	Help	
Vin	• 🗄 🔠 •	😂 🖬 🥔	X D B	1) - (1 -	- 💽 - [4 🖪 🔍 🌹		Debug
٢	📂 🐝 🗞	880	‰ 43 - 1	al 🖬 🏟	ш 🕨	60 SI (I S		Hex 🛛 🖾 🕯
÷	Arduino 1.0).x - Ardui	no Nanc 🝷 🤗	-	Со	ntinue (F5)		

4. Debug⇒Disable debugWIRE and Closeを選ぶことによってデバッグ動作を抜け出してください。

File Edit View VAssistX ASF Project Build	Deb	ug Tools Window Help	
🗛 - 🗉 🖅 🚅 🎜 🎜 🗛 🕰 🖉 - e		Windows	•
i 😨 🖾 🖓 🍋 🔓 😭 0, 🔬 🗳 🚽 💷 🔒	PII	Start Debugging and Break	Alt+F5
😳 Arduino 1.0.x 🝷 Arduino Nanc 👻 🥊		Stop Debugging	Ctrl+Shift+F5
Disassembly main con Blink con X Atm		Start Without Debugging	Ctrl+Alt+F5
		Disable debugWIRE and Close	
→ Binkcpp → C:\workspa	Þ.	Continue	FS
Blink	~	Execute Stimulifile	
Turns on an LED on for one second	4	Set Stimulifile	
This example code is in the publi		Restart	
*/	п	Break All	Ctrl+F5
#include "arduino.h"	63	QuickWatch	Shift+F9
(1.0° 42.1 1.00 1.1	91	Step Into	F11

警告 デバッグWIREを禁止することが重要です。

情報 デバッグWIREの禁止は目的対象をリセットしてDWENヒューズをリセットし、再びISPインターフェースを使うことができます。プログラム(0)されたDWENヒューズを持つことは、クロック系のいくつかの部分に全休止動作形態で動くことを許します。これは休止動作中にAVRの消費電力を増します。従ってデバッグWIREが使われない時にDWENヒューズは常に禁止されるべきです。

5. 基板をリセットしてLEDの点滅を観察してください。



5. ATmega328P応用

我々はA/D変換器(ADC)を使って光感知器を読み、I²Cインターフェースを使って温度感知器を読み、そしてSPIインターフェースを使ってSD カートドにデータを格納する簡単な応用を作成しつつあります。また、光感知器値と温度感知器値はmEDBG COMホートを通して送信され ます。

I²CについてはArduinoからの"Wire"ライブラリが必要とされ、SDカートについてはインクルートされるべき"SD library"が必要とされます。

5.1. コンパ イラ構成設定

1. プロジェクト(ここではATmega328P_1)を右クリックしてPropertiesを選んでください。

2. Toolchain⇒AVR/GNU C Compiler⇒Directories下に以下を追加してください(前もって追加したディレクトリが未だ一覧にあるべき です)。

C:\Program Files (x86)\Arduino\Hardware\Arduino\Eavr\libraries\Wire

C:\Program Files (x86)\Arduino\Hardware\Farduino\Favr\libraries\Wire\Futility

 $C: \label{eq:constraint} C: \ensuremath{\texttt{P}}\xspace{\texttt{$

C:\Program Files (x86)\Arduino\Ibraries\SD\src

C:\Program Files (x86)\Arduino\Ibraries\SD\src\u00e4utility

3. Toolchain⇒AVR/GNU C++ Compiler⇒Directoriesに対して繰り返してください。

警告 これらのパス追加中に"Relative Path"のチェック外しを確実にしてください。

Add Include Paths (-I)	
Include Paths (-I)	
Relative Path	
	OK Cancel

図5-1. Atmel Studio : Compiler Setup

Build Build Events	Configuration: Active (Debug)	Platform: Active (AVR)	
Toolchain	Configuration Manager		
Device Tool	 AVR/GNU Common General OutputFiles 	AVR/GNU C++ Compiler Directories Include Paths (-I)	a
	 AVR/GNU C Compiler General Preprocessor Symbols Directories Optimization Debugging Warnings Miscellaneous AVR/GNU C++ Compiler General Preprocessor Symbols Directories Optimization Debugging Warnings 	C:\Program Files (x86)\Arduino\hardware\arduino\avr\cores\arduino C:\Program Files (x86)\Arduino\hardware\arduino\avr\variants\standard C:\Program Files (x86)\Arduino\hardware\arduino\avr\variants\eightanaloginputs C:\Program Files (x86)\Arduino\libraries\SD\src\utility C:\Program Files (x86)\Arduino\hardware\arduino\avr\libraries\Wire C:\Program Files (x86)\Arduino\libraries\SD\src C:\Program Files (x86)\Arduino\libraries\SD\src C:\Program Files (x86)\Arduino\hardware\arduino\avr\libraries\SPI	

5.2. 依存ファイル追加

▶ 実施事項 ライブラリから実際のAtmel Studio プロジェクトへ.ccpソース ファイルを追加してください。

5.2.1. Wireライブラリに対して

- 1. プロジェクトを右クリックし、"Add⇒Existing Item…"へ行ってください。
- 2. 開いたダイアログ枠に於いて、閲覧部でC:¥Program Files (x86)¥Arduino¥hardware¥arduino¥avr¥libraries¥Wireを狙ってください。
- 3. Wire.ccpファイルを選択して 'As a link(リンクとして)' 追加してください。
- 4. "wire¥utility"副ディレクトリに対して同様に繰り返し、twi.cファイルを選択して'As a link(リンクとして)'追加してください。

Add Existing Item - ATmega	328P_1				X
子 🔵 🗢 👢 « Program Fil	es (x86) ► Arduino ► hardware ► arduino ►	avr 🕨 libraries 🕨 Wi	re 🕨 👻	← Search Wire	Q
Organize New folder					0
<table-cell-rows> Atmel Studio 6.2 👗</table-cell-rows>	Name	Date modified	Туре	Size	
1.2	👌 examples	25.02.2015 10:34	File folder		
=	🧑 utility	25.02.2015 10:34	File folder		
☆ Favorites	 keywords.txt 	08.02.2015 23:39	TXT File	1 KB	
💻 Desktop	 library.properties 	08.02.2015 23:39	PROPERTIES File	1 KB	
🐌 Downloads 👘	💓 Wire.cpp	08.02.2015 23:39	C++ Source File	8 KB	
laces Recent Places	👔 Wire.h	08.02.2015 23:39	H File	3 KB	
🔲 Desktop					
📜 Libraries					
Documents					
🔿 Git 🔻					
File name	: Wire.cpp		•	All Files (*.*)	•
				Add 🗸 Cano	el
				Add	
	General			Add As Link	

5.2.2. SDライフ・ラリに対して

- 1. プロジェクトを右クリックし、"Add⇒Existing Item…"へ行ってください。
- 2. 開いたダイアログ枠に於いて、閲覧部でC:¥Program Files (x86)¥Arduino¥hardware¥arduino¥avr¥libraries¥SPIを狙ってください。
- 3. SPI.ccpファイルを選択して 'As a link(リンクとして)' 追加してください。
- 4. C:¥Program Files (x86)¥Arduino¥libraries¥SD¥srcに対して同様に繰り返してください。
 - a. SD.cppとFile.cppのファイルを複数選択してください。
 - b. "SD¥utility"副ディレクトリに対して同様に繰り返し、SdVolume.cpp, Sd2Card.cpp, SdFile.cppを選択して'As a link(リンクとして)'追加してください。

organize New fold	der				•
•	Name	Date modified	Туре	Size	
Desktop	 FatStructs.h 	08.02.2015 23:39	H File	17 KB	
Libraries	💕 Sd2Card.cpp	08.02.2015 23:39	C++ Source File	21 KB	
Occuments Git	Ø Sd2Card.h	08.02.2015 23:39	H File	10 KB	
	Sd2PinMap.h	08.02.2015 23:39	H File	13 KB	
	🗿 SdFat.h	08.02.2015 23:39	H File	22 KB	
Subversion	Ø SdFatmainpage.h	08.02.2015 23:39	H File	9 KB	
	Ø SdFatUtil.h	08.02.2015 23:39	H File	3 KB	
Honrao, Rupali	👸 SdFile.cpp	08.02.2015 23:39	C++ Source File	42 KB	
Computer	Ø SdInfo.h	08.02.2015 23:39	H File	8 KB	
Network	👸 SdVolume.cpp	08.02.2015 23:39	C++ Source File	10 KB	
Control Panel					
Recycle Bin					
-					
File na	me: "SdVolume.cpp" "Sd2Card.cpp" "Sd2Card.cpp"""""""""""""""""""""""""""""""""""	5dFile.cpp"		✓ All Files (*.*)	

5.3. 応用開発

実行現在のBlink.cppファイルをsensors.cppファイルに改名し、LED点滅コート(即ち、setup()とloop())を削除してください。

実行 sensors.cppファイルの先頭でSDとWireのライブラリ用のヘッダ ファイルをインクルードください。

<pre>#include <wire.h> #include <sd.h></sd.h></wire.h></pre>	
▶ 実行 温度感知器に関する定義を追加してくだ	さい。
<pre>#define AT30TSE_TEMPERATURE_TWI_ADDR #define AT30TSE_TEMPERATURE_REG #define AT30TSE_TEMPERATURE_REG_SIZE #define AT30TSE_NON_VOLATILE_REG</pre>	0x4F 0x00 2 0x00
<pre>#define AT30TSE_CONFIG_RES_9_bit #define AT30TSE_CONFIG_RES_10_bit #define AT30TSE_CONFIG_RES_11_bit #define AT30TSE_CONFIG_RES_12_bit uint16_t resolution = AT30TSE_CONFIG_RE</pre>	0 1 2 3 2S_12_bit;
▶ 実行 使用ピンに対する定数を定義してください) o
<pre>const int analogInPin = A0; const int chipSelect = 10;</pre>	// アナログ入力ピン // SPI従装置選択ピン
▶ 実行 SDライブラリを使う変数を定義してください。	,
File myFile;	
▶ 実行 A/D変換結果と温度を格納する変数を定	至義してください。
<pre>double temp_result; int sensorValue = 0;</pre>	// A/D変換器A0からの読み込み値



```
実行 温度感知器を読む関数を追加してください。
```

```
uint16_t at30tse_read_register(uint8_t reg, uint8_t reg_type, uint8_t reg_size)
{
    uint8_t buffer[2], i=0;
    buffer[0] = reg | reg_type;
    buffer[1] = 0;
    /* AT30TSEの内部レシブスタ ポインタ */
    Wire.beginTransmission(AT30TSE_TEMPERATURE_TWI_ADDR);
    Wire.write(buffer[0]);
    Wire.endTransmission();
    Wire.requestFrom(AT30TSE_TEMPERATURE_TWI_ADDR, reg_size);
    while(Wire.available())
    {
        buffer[i] = Wire. read(); // 文字としてバ 小受信
        i++;
    }
    return (buffer[0] \ll 8) | buffer[1];
}
double at30tse_read_temperature()
{
    /* 16ビット温度レジスタ読み込み */
    uint16_t data = at30tse_read_register(AT30TSE_TEMPERATURE_REG, AT30TSE_NON_VOLATILE_REG,
                                          AT30TSE_TEMPERATURE_REG_SIZE);
    double temperature = 0;
    int8_t sign = 1;
    /* 負検査と符号ビット解除 */
    if (data & (1 << 15)) {
        sign *= -1;
        data &= ~(1 << 15);
    }
    /* 温度へ変換 */
    switch (resolution) {
        case AT30TSE_CONFIG_RES_9_bit:
            data = (data \gg 7);
            temperature = data * sign * 0.5;
            break;
        case AT30TSE_CONFIG_RES_10_bit:
            data = (data >> 6);
            temperature = data * sign * 0.25;
            break;
        case AT30TSE CONFIG RES 11 bit:
            data = (data >> 5);
            temperature = data * sign * 0.125;
            break;
        case AT30TSE_CONFIG_RES_12_bit:
            data = (data >> 4);
            temperature = data * sign * 0.0625;
            break;
        default:
            break;
    }
    return temperature;
```

Atmel

▶ 実行 setup()初期化関数を追加してください。

i

情報 Atumel Studioで関数名を書き始めると直ぐに下で示されるように可能な関数名が一覧にされます。

<pre>void setup() { se sessorValue Serial Setup Accept with: <tab> or <enter></enter></tab></pre>	<pre>void setup() { Serial.</pre>
<pre>bid setup() Serial.begin(9600); // 直列通信を if (!SD.begin(chipSelect)) { Serial.println("SD Card initializa return; } Serial.println("SD Card initialization SD.remove("data.txt"); Wire.begin();</pre>	r開く tion failed!"); done.");

```
実行 変数とloop()関数追加してください。
```

```
bool sensor_flg=0;
void loop() {
   sensorValue = analogRead(analogInPin); // A/D変換器(ADC)読み込み
temp_result = at30tse_read_temperature(); // 温度読み込み
    if (sensorValue>=500)
    {
       sensor_flg=1;
   }
   if (sensor_flg==1)
                                 // 特殊な感知器値でファイル内にデータ書き込み
    {
       sensor_flg=0;
       myFile = SD. open("data.txt", FILE_WRITE);
       if (myFile)
                                  // ファイルが開いていればそれに書き込み
        {
           myFile.print("sensor = " );
           myFile.print(sensorValue);
           myFile.print(" temp = ");
            myFile.print(temp_result);
           myFile.print("¥n");
           myFile.close(); // ファイルを閉じる
        }
       else
        {
           // ファイルが開かなかった場合は異常表示
           Serial.println("error opening data.txt");
        }
        // 読むために改めてファイルを開く
       myFile = SD. open("data.txt");
        if (myFile)
        {
           // 何も無くなるまでファイルから読み込み
           while (myFile.available()) {
               Serial.write(myFile.read());
            }
           // ファイルを閉じる
           myFile.close();
        } else {
           // ファイルが開かなかった場合は異常表示
           Serial.println("error opening data.txt");
        }
    }
    delay(500);
```

実行 ファイルを保存し、'Build'タブを選択するか、またはキーボードでF7キーを押すことによって解決策を構築してください。



 \checkmark

結果この構築は異常なしで成功裏に終了すべきです。

5.4. ハート・ウェア接続

光感知器と温度感知器はIO1 Xplained Pro基板上に存在します。

IO1 Xplained ProはAtmel Xplained Pro評価基盤に対する拡張基板です。それはマイクロSD、温度感知器、光感知器、その他色々を含むXplained Pro MCU基板に対して広く多様な機能を与えるように設計されます。IO1拡張基板は図5-2.で示されます。



ATmega328P Xplained MiniへのIO1 Xplained Pro基板の実際の接続は表5-1.で示されます。

Atmel

表5-1. 接紙	表5-1. 接続:IO1 Xplained Pro - ATmega328P Xpalined Mini				
IO1 Xplained Pro		ATmega328P Xplained Mini			
ピン番号	名前	ピン名	MCUピン		
3	ADC	PC0	A0		
11	TWI_SDA	PC4	SDA		
12	TWI_SCL	PC5	SCL		
2	GND	GND	-		
20	VCC	3V3	-		
15	SPI_SS_A	PB2	SPI_SS (D10)		
16	SPI_MOSI	PB3	SPI_MOSI		
17	SPI_MISO	PB4	SPI_MISO		
18	SPI_SCK	PB5	SPI_SCK		

5.4.1. 接続 : IO1 Xplained Pro - Arduino Xplained Pro

下図で示されるようにIO1 Xplained ProをArduino Xplained Proへ接続してください。



5.4.2. 接続 : ATmega328P Xplained Mini - Arduino Xplained Pro

次にATmega328P Xplained MiniをArduino Xplained Pro基板に接続することが必要です。Xplained Mini基板の列1からのPC5を Arduino Xplained ProのA5に(赤色接続)とXplained Mini基板の列11からのPC5をArduino Xplained ProのSCLに(緑色接続)行くように 接続してください。



右のように接続してください。



5.4.3. 接続 : USBケーフ ル

Ť

1

ATmega328P Xplained Mini基板にUSBケーブルを接続し、基板を図5-3.のように接続してください。

警告 ソケットにSDカードが正しく挿入されているのを確実にしてください。

情報 基板は光感知器が暗がりにないこの方法で置かれます。

情報 代替接続法:使用者は右下の図で示されるようにArduino Xplained Pro基板を使わずに9Pのオスーメス コネクタを使って IO1 Xplained Pro基板を接続することもできます。



5.5. 応用のデバッグ

次に応用をデバッグします。コードは特定条件(感知器値>500時)で 光感知器値と温度感知器値がSDカードに格納されるように書かれ ています。この条件に中断点(ブレークポイント)を配置して中断点に 当たることを調べてください。



実施事項 Debugを選んで'Start debugging and Break' をクリックしてください。

Deb	ug Tools Window Hel	>
	Windows	
M	Start Debugging and Break	Alt+F5
1	Stop Debugging	Ctrl+Shift+F5
Þ	Start Without Debugging	Ctrl+Alt+F5
	Disable debugWIRE and Cl	ose
	Continue	F5
G	Execute Stimulifile	
Ś	Set Stimulifile	
3	Restart	
62	QuickWatch	Shift+F9
•1	Step Into	F11
, 1	Step Over	F10
- III	Step Out	Shift+F11
•1	Run To Cursor	Ctrl+F10
1	Reset	Shift+F5
	Percepio Trace	
	Toggle Breakpoint	F9
	New Breakpoint	
9	Delete All Breakpoints	Ctrl+Shift+F9
	Clear All DataTips	
	Export DataTips	
	Import DataTips	
	Options and Settings	

警告 DWENヒューズが許可されていない場合に異常メッセージが表示されます。下で示されるように、Yes、をクリックしてヒューズを設定するのにAtmel StudioはISPを使います。

Launch Fail	:d
?	Failed to launch debug session with debugWIRE. This could be caused by reset line circuitry or disabled debugWIRE interface. Make sure that the reset line is free before continuing. Do you want to use SPI to enable the DWEN fuse?
	Yes No

結果 デバッガが開始され、mainで中断されます。今やデバッグを開始する準備が整いました。

施事項 Atmel StudioでView⇒Terminal Windowを選んでください。 mEDBG COMポート、Baud:9600を選び、"Connect"を 選んでください。

Terminal Window
Connect COM27 - Baud: 9600 - ASCII - 🔀 🖄 Save to file Options
Receive



情報 感知器データはmEDBG COMホートを通して受信されます。

実施事項 Solution Explorerからsensors.cppを開き、下図で示されるように中断点を配置してください。



🛜 実施事項 次に 🕨 印の選択、またはキーボートでのF5キー押下によってデバッグを始めてください。

結果端末(Terminal)ウィントゥに、SD Card initialization done、メッセージが現れるでしょう。

情報 光感知器はIO1 Xplained Pro上に存在し、感知器値は暗い時に増加し、明るい時に減少します。

実施事項 感知器値が増すように指で感知器を覆ってください。感知器値が増加されたために条件(sensorValue≧500)が真 なら、下で示されるようにプログラム実行が中断点(ブレークポイント)に行き当たるでしょう。



実施事項 ^{デ・バック} 中に感知器値を見るために、Debug⇒QuickWatch…を選んでください。式'sensorValue'を入力して'Add Watch' 釦を選んでください。

結果 監視1(Watch1)ウィンドウに'sensorValue'変数の値が表示されるでしょう。

ATmega_328P_1 (Debugging) - AtmelStudio File Edit View VAssistX ASF Project Build Debug Tools Window HI Where Image: State of the	Help → ₩ Debug → Ø spi_init → Q 3 № 5 ■ 1 Hex 8 → - : @ M 1 @ 9 5 - : @ # ATmega:
Disassembly CDC.cpp wiring.c main.cpp sensors.cpp × Arduit → loop.if ▼ ● if (sensorValue>=500)	ino.h ATmega_328P_1
<pre>void loop() { sensorValue = analogRead(analogInPin); // rea temp_result = at30tse_read_temperature(); // rea if (sensorValue>=500) { sensor_flg=1; } if (sensor_flg==1) // write data in file on p { sensor_flg=0; myFile = SD.open("data.txt", FILE_WRITE); if (myFile) // if the file opened okay, wr { myFile.print("sensor = "); myFile.print(sensorValue); } // sensorValue); // sensorValue/// sensorValue/// sensorValue/// sensorValue//// sensorValue////////////////////////////////////</pre>	Expression: sensorValue Value Value: Close Help
Watch 1 Name	Value 516

実施事項 'Watch1'ウィンドウと'QuickWatch'ウィンドウを閉じてください。

実施事項 キーボードでF5を押すことによって実行を続けてください。

結果 感知器の値はSDカートに格納されmEDBG COMホートを通して送信もされます。端末(Terminal)ウィントウは光感知器値と 温度値を示すでしょう。

図5-4. Atmel Studio : 端末(Terminal)ウィント・ウ

eminal Window	* 0 ×
🗭 Disconnect COM27 📲 Baud: 9600 📲 ASCII 🖹 🗙 😤 🛄 Save to file Options	
Receive	
SD Card initialization done.	A.
Sensor - 510 Cemp - 24.00	
	*
Send History	
Send	
	ASCII - LF CR Send
mEDRG Virtual COM Part (COM27)	
IEDBO VIItual COM POR (COM27)	

🖹 実施事項 Debug⇒Disable debugWIREを選ぶことによってデバッグ動作を向けだしてください。

警告 デバッグWIREを禁止することが重要です。

Atmel StudioでATmega328P Xplained MiniでのArduinoプロジェハを成功裏に追加しました。

追補A. 課題4に対する完全な解決策

sensors.cpp7r1N

```
#include "arduino.h"
#include <Wire.h>
#include <SD. h>
#define AT30TSE_TEMPERATURE_TWI_ADDR 0x4F
#define AT30TSE_TEMPERATURE_REG
                                      0x00
#define AT30TSE_TEMPERATURE_REG_SIZE 2
#define AT30TSE_NON_VOLATILE_REG
                                      0x00
#define AT30TSE CONFIG RES 9 bit
                                      0
#define AT30TSE CONFIG RES 10 bit
                                      1
#define AT30TSE_CONFIG_RES_11_bit
                                      2
                                      3
#define AT30TSE_CONFIG_RES_12_bit
uint16_t resolution = AT30TSE_CONFIG_RES_10_bit;
double temp_result;
int sensorValue = 0;
                             // A/D変換器(ADC) A0から読んだ値
const int analogInPin = A0; // アナロケ入力ピン
const int chipSelect = 10;
// SDユーティリティライブラリ関数を使うための変数を準備
File myFile;
void setup()
ł
                             // 直列通信を開く
   Serial.begin(9600);
   if (!SD. begin(chipSelect)) {
       Serial.println("SD Card initialization failed!");
       return;
   Serial.println("SD Card initialization done.");
   SD. remove("data.txt");
   Wire.begin();
}
uint16_t at30tse_read_register(uint8_t reg, uint8_t reg_type, uint8_t reg_size)
ł
   uint8_t buffer[2], i=0;
   buffer[0] = reg | reg_type;
   buffer[1] = 0;
   /* AT30TSEでの内部レシブスタ ホプンタ */
   Wire. beginTransmission(AT30TSE_TEMPERATURE_TWI_ADDR);
   Wire.write(buffer[0]);
   Wire.endTransmission();
   Wire.requestFrom(AT30TSE_TEMPERATURE_TWI_ADDR, reg_size);
   while(Wire.available())
       buffer[i] = Wire. read();
       i++;
   }
   return (buffer[0] \langle \langle 8 \rangle | buffer[1];
```

Atmel

```
#include <Wire.h>
double at30tse_read_temperature()
{
    /* 16ビット温度レジスタ読み込み */
    uint16_t data = at30tse_read_register(AT30TSE_TEMPERATURE_REG, AT30TSE_NON_VOLATILE_REG,
                                          AT30TSE_TEMPERATURE_REG_SIZE);
    double temperature = 0;
    int8_t sign = 1;
    /* 負検査と符号ビット解除 */
    if (data & (1 << 15)) {
        sign *= -1;
        data &= ^{\sim}(1 \iff 15);
    }
    /* 温度に変換 */
    switch (resolution) {
        case AT30TSE_CONFIG_RES_9_bit:
            data = (data >> 7);
            temperature = data * sign * 0.5;
            break;
        case AT30TSE_CONFIG_RES_10_bit:
            data = (data >> 6);
            temperature = data * sign * 0.25;
            break;
        case AT30TSE_CONFIG_RES_11_bit:
            data = (data >> 5);
            temperature = data * sign * 0.125;
            break;
        case AT30TSE_CONFIG_RES_12_bit:
            data = (data >> 4);
            temperature = data * sign * 0.0625;
            break;
        default:
            break;
    return temperature;
```

```
bool sensor_flg=0;
void loop() {
    sensorValue = analogRead(analogInPin); // A/D変換器(ADC)読み込み
temp_result = at30tse_read_temperature(); // 温度読み込み
    if (sensorValue>=500)
    {
        sensor_flg=1;
    }
    if (sensor_flg==1)
                       // 特定感知器値をファイルに書き込み
    {
        sensor_f1g=0;
        myFile = SD. open("data.txt", FILE_WRITE);
                             // ファイルが開いていれば、それに書き込み
        if (myFile)
        {
            myFile.print("sensor = ");
            myFile.print(sensorValue);
            myFile.print(" temp = ");
            myFile.print(temp_result);
            myFile.print("¥n");
            myFile.close(); // ファイルを閉じる
        }
        else
        {
            // ファイルが開かなければ異常を表示
            Serial.println("error opening data.txt");
        }
        // 読むために再びファイルを開く
        myFile = SD. open("data.txt");
        if (myFile)
        {
            // 何も無くなるまでファイルから読み込み
            while (myFile.available()) {
                Serial.write(myFile.read());
            }
            // ファイルを閉じる
            myFile.close();
        } else {
           // ファイルが開かなければ異常を表示
            Serial.println("error opening data.txt");
        }
    delay(500);
```

Atmel

<mark>6</mark>. 結び

この実施訓練で以下の作業が達成されました。

- Atmel StudioへのArduino風コート 書き形式支援を追加
- Arduinoプロジェクトヘデベッグ能力を追加
- どんなコート 変更もなしにAtmel Xplained MiniとAtmel StudioでのArduinoプロジェクトを置く
- もっと込み入った複雑なプログラムを書くことができる能力を追加

7. 改訂履歴

文書改訂	日付	注釈
42439A	2015年8月	初版文書公開

Atmel | Enabling Unlimited Possibilities[®]



Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA TEL:(+1)(408) 441-0311 FAX: (+1)(408) 436-4200 | www.atmel.com

© 2015 Atmel Corporation. / 改訂:Atmel-42439A-From-Maker-to-Manufacture-Bridging-the Gap-from-Arduino-to-AVR_TrainingManual_082015

Atmel[®]、Atmel^{ロゴ}とそれらの組み合わせ、AVR[®]、Enabling Unlimited Possibilities[®]とその他は米国と他の国に於けるAtmel Corporationの登録商標 または商標です。Windows[®]は米国及び他の国に於けるMicrosoft Corporationの登録商標です。他の用語と製品名は一般的に他の商標です。

お断り:本資料内の情報はAtmel製品と関連して提供されています。本資料またはAtmel製品の販売と関連して承諾される何れの知的所有権も禁 反言あるいはその逆によって明示的または暗示的に承諾されるものではありません。Atmelのウェブサイトに位置する販売の条件とAtmelの定義での 詳しい説明を除いて、商品性、特定目的に関する適合性、または適法性の暗黙保証に制限せず、Atmelはそれらを含むその製品に関連する暗示 的、明示的または法令による如何なる保証も否認し、何ら責任がないと認識します。たとえAtmelがそのような損害賠償の可能性を進言されたとし ても、本資料を使用できない、または使用以外で発生する(情報の損失、事業中断、または利益と損失に関する制限なしの損害賠償を含み)直 接、間接、必然、偶然、特別、または付随して起こる如何なる損害賠償に対しても決してAtmelに責任がないでしょう。Atmelは本資料の内容の正 確さまたは完全性に関して断言または保証を行わず、予告なしでいつでも製品内容と仕様の変更を行う権利を保留します。Atmelはこに含まれた 情報を更新することに対してどんな公約も行いません。特に別の方法で提供されなければ、Atmel製品は車載応用に対して適当ではなく、使用さ れるべきではありません。Atmel製品は延命または生命維持を意図した応用での部品としての使用に対して意図、認定、または保証されません。

安全重視、軍用、車載応用のお断り: Atmel製品はAtmelが提供する特別に書かれた承諾を除き、そのような製品の機能不全が著しく人に危害を 加えたり死に至らしめることがかなり予期されるどんな応用("安全重視応用")に対しても設計されず、またそれらとの接続にも使用されません。安全 重視応用は限定なしで、生命維持装置とシステム、核施設と武器システムの操作用の装置やシステムを含みます。Atmelによって軍用等級として特に明確 に示される以外、Atmel製品は軍用や航空宇宙の応用や環境のために設計も意図もされていません。Atmelによって車載等級として特に明確に示 される以外、Atmel製品は車載応用での使用のために設計も意図もされていません。

© HERO 2021.

本応用記述はAtmelのAN12077応用記述(Rev.42439A-08/2015)の翻訳日本語版です。日本語では不自然となる重複する形容表現は省略されて いる場合があります。日本語では難解となる表現は大幅に意訳されている部分もあります。必要に応じて一部加筆されています。頁割の変更によ り、原本より頁数が少なくなっています。

必要と思われる部分には()内に英語表記や略称などを残す形で表記しています。

青字の部分はリンクとなっています。一般的に赤字の0,1は論理0,1を表します。その他の赤字は重要な部分を表します。