Hier eine kurze Anleitung zur Ergänzung des Artikels STM32 - Einstieg mit Em::Blocks auf Mikrocontroller.net http://www.mikrocontroller.net/articles/STM32 - Einstieg mit Em::Blocks

Inhalt: Schritte zur Erzeugung von *.hex / *.bin Dateien zum direkten Übertragen des Programms in das STM32F4 Discovery-Board. Der Aufruf erfolgt über das Kommandozeilentool ST-LINK direkt aus der EMBlocks IDE heraus.

START:

Schritt 1: Rechtsklick auf Projekt → Properties...



Schritt 2: Unter Build targets \rightarrow Generate Hex file

Project settings	Build targets	Build scripts Notes C		C/C++ parser options	EnvVars options			
Build targets		Select	ed build tar	get options				
	Add	Type:		Executable	•			
	Rename			🔽 Generate Hex file				
	Duplicate			This option is for the external Hex conversion tool. For toolchain linkers with native hex-support check the linker settings.				
	Delete	Output	t filename:	bin\Debug\STM32F4	Dev.elf			
		Objects	s output dir	; obj\Debug\				
	Virtual targets.	Build t	Build target files:					
	Dependencies.		ISIS\Includ	.h	-			
	Re-order							
	Build options	CM	ISIS\Includ	e\core_cm4_simd.h				
		V CN	CMSIS (Include (core_cm)-unc.n CMSIS (Include (core_cm)-unc.n					
		🖌 ST	✓ STM32F4xx_StdPeriph_Driver\inc\misc.h					
		✓ ST	M32F4xx_9	StdPeriph_Driver\inc\st	m32f4xx_adc.h			
	Construction	J I ST	M32F4xx_9	StdPeriph_Driver\inc\st	m32f4xx_can.h	÷		
	from target	Togg	le checkma	rks	Sele	cted file properties		

Schritt 3: Tools \rightarrow Configure tools ...

File Edit View Search Project Build Debug	Tools Plugins DoxyBlocks	Settings Help
: 🖻 😂 🗔 🥔 I 🤊 🗠 I 🕉 🗈 🛆 🔍	STM32F4 ST LINK	😂 🕮 📖 🔆 Debug 🔹 🕴
Management	Configure tools	
Workspace	64 /* Priv	ate functions

Schritt 4: →Add

STM32F4 ST LINK	Move up
	Move down
	Add
	Add separator
	Edit
	Remove

Schritt 5: folgende Daten eintragen:

Name: STM32F4 ST LINK

Executable: C:\Program Files\STMicroelectronics\STM32 ST-LINK Utility\ST-LINK Utility\ST-LINK_CLI.exe

 Parameters:
 -c SWD -P "\${TARGET_OUTPUT_DIR}\\${TARGET_OUTPUT_BASENAME}.hex" 0x08000000 -V

 "\${TARGET_OUTPUT_DIR}\\${TARGET_OUTPUT_BASENAME}.hex" -Rst -Run

Working directory: \${PROJECT_DIR}

Durch diese Einstellungen wird das *.hex File aus dem Debug Ordner in den Controller übertragen und anschließend nochmals verifiziert. Danach führt durch die Parameter RST/RUN der Controller das Programm aus.



Schritt 5: Das wars schon! Unter Tools das STM32F4 ST LINK Tool ausführen (siehe Bild aus Schritt 3), anschließend wird das ST Link Tool ausgeführt. Bei kleinen Programmen ist es möglich, dass das folgende Fenster nur wenige Sekunden aufpoppt und dadurch Fehler z.B. durch falsche Einstellungen nicht gesehen werden. Durch ankreuzen von "Launch tool hidden with standard output redirected" in Schritt 5 kann man die Infos auch in der IDE ausgeben lassen.



Alternativer Schritt:

Wenn man statt eines *.hex ein *.bin File benötigt, kann man auch folgende Einstellungen unter Build options \rightarrow Pre/post build steps tätigen und die Auswahl auf Always stellen:

arm-none-eabi-objcopy -O binary \${TARGET_OUTPUT_DIR}\\${TARGET_OUTPUT_BASENAME}.elf \${TARGET_OUTPUT_DIR}\\${TARGET_OUTPUT_BASENAME}.bin

Project build options		• ×
STM32F4Dev	Selected compiler	
Debug	ARM GCC Compiler (EmBlocks - bare-metal)	•
	Device Compiler settings Linker settings Assembler settings Search directories Pre/post build steps Custom variables	
	Pre-build steps	
	Disabled	
		^
		*
		P.
	Post-build steps	
	Always	-
	arm-none-eabi-objcopy -O binary \${TARGET_OUTPUT_DIR}\\${TARGET_OUTPUT_BASENAME}.elf \${TARGET_OUTPUT_DIR}\\${TARGET_OUTPUT_BASENAME}.bin	*
		-
	4	. F.
	Cancel	
	108 USART Initialisierung();	

Dadurch wird ein *.bin File im Anschluss erstellt. Ab hier kann Schritt 4 (siehe oben) durchgeführt werden, wobei dort die *.hex gegen *.bin ausgetauscht werden müssen.

Happy Coding!!!