# Erzeugen neues Target

* RTOS + ETHERNET

D:\GitHub\mbed\tools>build.py -m NUCLEO\_F767ZI -t GCC\_ARM -r –e

D:\GitHub\mbed\tools>make.py -m NUCLEO\_ F767ZI -t GCC\_ARM -n NET\_6

* RTOS

D:\GitHub\mbed\tools>build.py -m NUCLEO\_F446RE -t GCC\_ARM -r

D:\GitHub\mbed\tools>make.py -m NUCLEO\_ F446RE -t GCC\_ARM -n CMSIS\_RTOS\_2

## Testen:

Auf Target kopieren:

"D:\GitHub\mbed\.build\test\NUCLEO\_F767ZI\GCC\_ARM\NET\_6\ udp\_client.bin"

Auf Server (PC) starten:

"D:\GitHub\mbed\tools\host\_tests\udpecho\_server.py" – vorher Server IP Adresse anpassen

* RTOS

Auf Target kopieren

"D:\GitHub\mbed\.build\test\NUCLEO\_F446RE\GCC\_ARM\CMSIS\_RTOS\_1\basic.bin"

### Ausgabe bei Erfolg

* RTOS + ETHERNET

Im Terminal

{{timeout;20}}  
{{host\_test\_name;udpecho\_client\_auto}}  
{{description;UDP echo client}}  
{{test\_id;NET\_6}}  
{{start}}  
MBED: UDPCllient waiting for server IP and port...  
192.168.1.11:7195  
MBED: Address received: 192.168.1.11:7195  
IP: 192.168.1.13  
MBED: UDPClient IP Address is 192.168.1.13  
{{success}}  
{{end}}

Auf Server:

client: ('192.168.1.13', 49153)

data: ez{@>>6@qQwG[JuDr1.w!&vB9%B,9cKfB1ZZ2xL5AvzzYSY9sp,PEyXa8Ipf<e<r

client: ('192.168.1.13', 49153)

data: @RSTQ5qUya])2J;PB=M=fcJv?ERMd0sKz{1S$@8`k&&zA!"o2}oX{ZU?/+]-D:oZ

client: ('192.168.1.13', 49153)

data: &Ky-(dUJ,QOvyPlu):>[32a@;1zU-Z$qGjbAX&-.2Rtc\*w8aHx\_j+)VtcF<4E0051](mailto:32a@;1zU-Z$qGjbAX&-.2Rtc*w8aHx_j+)VtcF%3c4E0051)

…

data: PWTbteO$.=4cewal{%.lIBv!)!sVv4kvlj+NT{&JYT@6L+TY5t:<;J)Z'[3]!4/m

client: ('192.168.1.13', 49153)

data: {{success}}

{{end}}

* RTOS

Th 1: 0  
Th 1: 1  
Th 1: 0  
Th 1: 1  
Th 1: 0

## Export to gcc\_arm Project

* RTOS + ETHERNET

D:\GitHub\mbed\tools>project.py -m NUCLEO\_ F767ZI -i gcc\_arm -n NET\_6 -b

Auspacken nach

D:\GitHub\mbed\.build\export\NET\_6\_gcc\_arm\_NUCLEO\_ F767ZI

* RTOS

D:\GitHub\mbed\tools>project.py -m NUCLEO\_F446RE -i gcc\_arm -n CMSIS\_RTOS\_2 –b

Auspacken nach

D:\GitHub\mbed\.build\export\CMSIS\_RTOS\_2\_gcc\_arm\_NUCLEO\_F446RE

Makefile anpassen:

OBJECTS = ./main.o ./env/test\_env.o

Vorher war die Liste mehrzeilig – sehr lang (evt. Fehler im mbed Projektexporter)

### Testen

Make D:\GitHub\mbed\.build\export\NET\_6\_gcc\_arm\_NUCLEO\_ F767ZI\NET\_6>make

Ausgabe

arm-none-eabi-objcopy -O binary NET\_6.elf NET\_6.bin

arm-none-eabi-size NET\_6.elf

text data bss dec hex filename

105780 14748 43212 163740 27f9c NET\_6.elf

Auf Target kopieren:

"D:\GitHub\mbed\.build\export\NET\_6\_gcc\_arm\_NUCLEO\_F767ZI\NET\_6\NET\_6.bin"

### Ausgabe bei Erfolg

Siehe 1.1.1

## Anpassen des Makefile I

Lib manager hinzufügen:

AR = $(GCC\_BIN)arm-none-eabi-ar

Neue Regel hinzufügen:

lib:

$(AR) -ru D:\GitHub\mbed\.build\export\ NET\_6\_gcc\_arm\_NUCLEO\_F767ZI\NET\_6\mbed\TARGET\_NUCLEO\_F767ZI\ TOOLCHAIN\_GCC\_ARM\libmbed.a $(SYS\_OBJECTS)

Ausführen:

make lib

Ausführen:

make clean

Anpassen Makefile

SYS\_OBJECTS =

Ausführen:

make

Ergebnis:

arm-none-eabi-objcopy -O binary NET\_6.elf NET\_6.bin

arm-none-eabi-size NET\_6.elf

text data bss dec hex filename

105748 14748 43212 163708 27f7c NET\_6.elf

Testen:

Ergebnisse s.oben