Biwak - Feature #218

Clean up CAN code

17.08.2022 16:17 - Maximilian Seesslen

Status:	Erledigt	**Beginn:**	17.08.2022
Priorität:	Normal	**Abgabedatum:**	
Zugewiesen an:	Maximilian Seesslen	**% erledigt:**	0%
Kategorie:		**Geschätzter Aufwand:**	0.00 Stunde
Zielversion:	v2.1.0	**Aufgewendete Zeit:**	0.00 Stunde
CS Zielversion:			

Beschreibung

Clean up all the timing calculation garbage.
The values should be determined only by the periphery clock and bus Speed.

This has to be done for FDCAN and classic CAN.

Check if CFdCan and CCan can be merged. Filter is a little bit different. There are already different HAL-implementations for Classic-CAN.

Fuer 13/2: prescaler=Clock/speed/16; (Clock/speed)%16 muss 0 sein
48Mhz, 125Kbit, prescaler: 24
48Mhz, 500Kbit, prescaler: 6
8Mhz, 500Kbit, prescaler: 1

Bei hohen raten und geringem Takt funktioniert 13/2 nicht mehr

Fuer 10/1; prescaler=Clock/speed/12; (Clock/speed)%12 muss 0 sein
48Mhz, 2000Kbit, prescaler: 2

Fuer 6/1; prescaler=Clock/speed/8; (Clock/speed)%8 muss 0 sein
16Mhz, 2000Kbit, prescaler: 2

divider=1+(SEG1+SEG2)

```
struct SConfigs
{
    char SEG1;
    char SEG2;
} configs[]=
{
   {13,2},
   {10,1},
   {8,1},
   {6,1},
};

int getConfig(speed)
{
   for(int i1=0; i1<sizeof(configs)/sizeof(configs[0]); i1++)
   {
      if( (HAL_RCC_CANClock /getBitRate(speed) ) % (configs[i1].SEG1 + configs[i1].SEG2 + 1) ) ==
0 )
      {
         return(i1);
      }
   }
   return(-1);
}

config=getConfig(enum);
lAssert( (config>=0) );
```

```
bool getConfig(clock, speed, &seg1, &seg2, &preescaler)
{
   seg1=13; seg2=2
   // Because prescaler=clock/speed/divider we need an divider that produces an integer.

// Don't waste much time; only check these combinations: { {13,2}, {10,1}, {8,1}, {6,1} }, hopeful
ly the first one fits
   while(seg1>=6)
   {
      int divider=seg1+seg2+1;
      if( (clock /getBitRate(speed) ) % divider ) == 0 )
      {
         prescaler=clock/speed/divider;
         return(true);
      }
      if(seg2>=2)
         {seg2--; seg1--;}
      seg1-=2;
   }
   return(false);
}
```

Specified: 1Mbit, 500kBit, 250kBit, 125kBit, 50kBit, 20kBit, 10kBit, 5kBit

```
enum {
   e1MBit = 1,
   e500KBit = 2,
   e250KBit = 4,
   e125KBit = 8,
   e50KBit = 20,
   e20KBit = 50,
   e10KBit = 100,
   e5KBit = 200,
}
int getBitRate(enum)
{
   return( (1000/enum) * 1000 );
}
```

Speed = (1000/enum)*1000

The clock tree is created in the libstm at the moment.
There can be aliases in "stm32/rcc.h" for each family.

```
uint32_t HAL_RCC_GetCANFreq(void) asm("HAL_RCC_GetPCLK1Freq");
```

Historie

#1 - 17.08.2022 16:47 - Maximilian Seesslen

- Beschreibung aktualisiert

#2 - 17.08.2022 17:44 - Maximilian Seesslen

- Beschreibung aktualisiert

#3 - 17.08.2022 17:54 - Maximilian Seesslen

- Beschreibung aktualisiert

#4 - 17.08.2022 18:06 - Maximilian Seesslen

- Beschreibung aktualisiert

#5 - 17.08.2022 18:10 - Maximilian Seesslen

- Beschreibung aktualisiert

#6 - 17.08.2022 18:21 - Maximilian Seesslen

- Beschreibung aktualisiert

#7 - 17.08.2022 18:39 - Maximilian Seesslen

- Beschreibung aktualisiert

#8 - 17.08.2022 18:40 - Maximilian Seesslen

- Beschreibung aktualisiert

#9 - 17.08.2022 18:50 - Maximilian Seesslen

- Beschreibung aktualisiert

#10 - 27.08.2022 12:50 - Maximilian Seesslen

- Status wurde von Neu zu Erledigt geändert