G. Pillmann

20.09.2013

Schnelle diskrete Hartley- und Fourier-Transformation *Fast discrete Hartley and Fourier transformation*

Es wird das Propeller-Objekt **Hartley.Spin** mit folgenden Eigenschaften beschrieben: *In the following, the propeller Hartley.Spin object is described with the following properties:*

- Transformation mit 8, 16, 32 .. 2048.. Punkten *Transformation with 8, 16, 32 .. 2048 .. points*
- aufteilbar auf 1, 2 oder 4 Cogs *Assignable to 1, 2 or 4 Cogs*
- Eingangsdaten 15 Bit plus Vorzeichen bzw. 16 Bit ohne Vorzeichen *Input data 15 bits plus sign or 16-bit unsigned*
- Ausgangsdaten 31 Bit plus Vorzeichen *Output data 31 bits plus sign*
- Fourier-Spektrum für 1024 Punkte bei 80 MHz Prozessortakt und 4 Cogs in 7.5 ms *Fourier spectrum for 1024 points with 80 MHz processor clock and 4 cogs in 7.5 ms*

Mathematische Grundlagen für das Propeller-Objekt sind: *Mathematical foundations for the Propeller object are:*

- Schnelle diskrete Hartley-Transformation (DHT) zur Berechnung der diskreten Fourier-Transformation (DFT) mittels Rekursion *Fast Discrete Hartley Transform (DHT) for calculating the discrete Fourier transform (DFT) by means of recursion*
- Cordic-Verfahren für Sinus, Cosinus, Multiplikation, Betrag, Phase *CORDIC technique for sine, cosine, multiplication, magnitude, phase*

Definitionen *Definitions*

Für die reelle Datenfolge *For the real data sequence*

```
q(t)
```

und die Indexbereiche *and index ranges*

```
t = 0..n-1
f = 0..n-1
```

mit *with*

```
n = 2^p und p >= 0
```

ist die diskrete Hartley-Transformierte `H(f)` über die Gleichung *the discrete Hartley transform H (f) is defined by the equation*

```
H(f) = 1/n × Σ{t=0..n-1}{q(t)×cos(360°ft/n) + q(t)×sin(360°ft/n)}          [Gl.01]
```

definiert. Aus der reellen Hartley-Transformierten H(f) ergibt sich die komplexe Fourier-Transformierte F(f) zu *The complex Fourier transform F (f) is calculated from real Hartley transform H (f) according to the equation*

```
F(f) = (H(f)+H(n-f))/2 + j×(H(f)-H(n-f))/2                                     [Gl.02]
```

mit *with*

```
H(n-0) = H(0)
```

Die inverse diskrete Hartley-Transformierte q(t) ist über die Gleichung *The inverse discrete Hartley transform q(t) is defined by the equation*

```
q(t) = 1/n × Σ{f=0..n-1}{H(f)×cos(360°ft/n) + H(f)×sin(360°ft/n)}             [Gl.03]
```

definiert. Sie kann mit demselben Algorithmus wie die DHT berechnet werden. *It can be calculated with the same algorithm as the DHT.*

Schnelle Berechnung der DHT (Prinzip) *Fast calculation of DHT (Principle)*

Die reelle Datenfolge *The real data sequence*

```
q(t)   = {a0 b0 a1 b1 a2 b2 a3 b3 .. a(n/2-1) b(n/2-1)}
```

kann zum einen durch Addition der Folgen *can be generated firstly by adding the sequences*

```
qa0(t) = {a0 0 a1 0 a2 0 a3 0 .. a(n/2-1) 0}
q0b(t) = {0 b0 0 b1 0 b2 0 b3 .. 0 b(n/2-1)}
```

und zum anderen durch einfaches Mischen der Folgen *secondly by simple mixing of the sequences*

```
qa(t)  = {a0 a1 a2 a3 .. a(n/2-1)}
qb(t)  = {b0 b1 b2 b3 .. b(n/2-1)}
```

erzeugt werden. Die DHT der Folgen qa und qb seien *The DHT of the data sequences qa and qb may be*

```
Ha(f)  =  {A0 A1 A2 A3 .. A(n/2-1)}
Hb(f)  =  {B0 B1 B2 B3 .. B(n/2-1)}
```

Nach dem Dehnungstheorem der Hartley-Transformation haben die Folgen *According to the expansion theorem of Hartley transformation the sequences*

```
qa0(t) = {a0 0 a1 0 a2 0 a3 0 .. a(n/2-1) 0}                                   [Gl.04]
qb0(t) = {b0 0 b1 0 b2 0 b3 0 .. b(n/2-1) 0}
```

die DHT *have the DHT*

```
Ha0(f) = {A0 A1 A2 A3 … A(n/2-1) | A0 A1 A2 A3 … A(n/2-1)} = { Ha(f') | Ha(f') }
Hb0(f) = {B0 B1 B2 B3 … B(n/2-1) | B0 B1 B2 B3 … B(n/2-1)} = { Hb(f') | Hb(f') }
```

mit *with*

```
f  = 0..n-1
f' = 0..n/2-1
```

Werden die Elemente der Folge qb0(t) um eine Stelle nach rechts zyklisch verschoben, entsteht die Folge *If the elements of the sequence qb0(t) are cyclically shifted by one position to the right the data sequence*

```
q0b(t) = {0 b0 0 b1 0 b2 0 b3 .. 0 b(n/2-1)}
```

Diese hat nach dem Verschiebungstheorem der Hartley-Transformation die DHT *is created. According to the shift theorem of Hartley transformation it has the DHT*

```
H0b(f) = Hb0(f)×cos(360°/n ×f) + Hb0(n-f)×sin(360°/n ×f)                [Gl.05]
```

Nach dem Additionstheorem der Hartley-Transformation hat daher die Folge *According to the addition theorem of Hartley transformation the data sequence*

```
q(t)   = qa0(t) + q0b(t)                                                [Gl.06]
```

die DHT *has the DHT*

```
H(f)   =   Ha0(f ) + H0b(f )
       =   Ha0(f ) + Hb0(f )×cos(360°/n ×f  ) + Hb0(n -f )×sin(360°/n ×f  )
       = { Ha (f') + Hb (f')×cos(360°/n ×f' ) + Hb (n'-f')×sin(360°/n ×f' ) |
           Ha (f') + Hb (f')×cos(360°/n ×f'') + Hb (n'-f')×sin(360°/n ×f'') }
```

mit *with*

```
n'     = n/2
f      = 0..n-1
f'     = 0..n/2-1
f''    = n/2..n-1
Hb(n') = Hb(0)
```

Der Berechnungsaufwand für H(f) wächst quadratisch mit der Länge n. Da für Ha0(f) und H0b(f) nur die halbsolangen Ha(f) und Hb(f) berechnet werden müssen, und die Multiplikationen mit den Cosinus- und Sinus-Werten den durch die Halbierung erzielten Gewinn bei weitem nicht zunichte machen, führt wiederholtes Anwenden obigen Vorgehens zu einer Reduzierung des Berechnungsaufwand für H(f) von O(n × n) auf O(n × log(n)). *The computational effort for H(f) grows quadratically with the length n. As for Ha0(f) and H0b(f) only the half-length Ha(f) and Hb(f) must be calculated. Repeatedly applying the above procedure leads to a reduction of the computational effort for H(f) from O(n × n) to O(n × log(n)).*

Schnelle Berechnung der DHT (Optimierung) *Fast calculation of DHT (Optimization)*

Für die Folge mit 1 Element (`n=1,t=0,f=0`) *For the sequence with 1 element*

```
q(t) = {q0}
```

ist die DHT *is the DHT*

```
H(f) = {H0}
```

mit *with*

```
H0 = q0
```

Für die Folge mit 2 Elementen (`n=2,t=0..1,f=0..1`) *For the sequence with 2 elements*

```
q(t) = {q0 q1} = {a0 b0} = {a0 0} + {0 b0}
```

berechnet sich die DHT mit *the DHT is calculated with*

```
Ha0(f) = {A0 | A0}
Hb0(f) = {B0 | B0}
```

und *and*

```
{A0} = Ha(f')
{B0} = Hb(f')
```

zu *to*

```
H(f) = {H0 H1}
```

mit *with*

```
H0 = A0 + B0×cos(  0°) + B0×sin(  0°)
H1 = A0 + B0×cos(180°) + B0×sin(180°)
```

Dies kann vereinfacht werden zu *This can be simplified to*

```
H0 = A0 + B0×cos(  0°) + B0×sin(  0°) = A0 + B0 = q0 + q1     (Cog0a)
H1 = A0 - B0×cos(  0°) + B0×sin(  0°) = A0 - B0 = q0 - q1     (Cog0a)
```

Für die Folge mit 4 Elementen (`n=4,t=0..3,f=0..3`) *For the sequence with 4 elements*

```
q(t) = {q0 q1 q2 q3} = {a0 b0 a1 b1} = {a0 0 a1 0} + {0 b0 0 b1}
```

berechnet sich die DHT mit *the DHT is calculated with*

```
Ha0(f) = {A0 A1 | A0 A1}
Hb0(f) = {B0 B1 | B0 B1}
```

und *and*

```
{A0 A1} = Ha(f')
{B0 B1} = Hb(f')
```

zu *to*

```
H(f) = {H0 H1 H2 H3}
```

mit *with*

```
H0 = A0 + B0×cos(  0°) + B0×sin(  0°)
H1 = A1 + B1×cos( 90°) + B1×sin( 90°)
H2 = A0 + B0×cos(180°) + B0×sin(180°)
H3 = A1 + B1×cos(270°) + B1×sin(270°)
```

Dies kann vereinfacht werden zu *This can be simplified to*

```
H0 = A0 + B0×cos(  0°) + B0×sin(  0°)  = A0 + B0 = q0 + q2 + q1 + q3     (Cog0a)
H1 = A1 + B1×cos( 90°) + B1×sin( 90°)  = A1 + B1 = q0 - q2 + q1 - q3     (Cog0a)
H2 = A0 - B0×cos(  0°) - B0×sin(  0°)  = A0 - B0 = q0 + q2 - q1 - q3     (Cog0a)
H3 = A1 - B1×cos( 90°) - B1×sin( 90°)  = A1 - B1 = q0 - q2 - q1 + q3     (Cog0a)
```

Für die Folge mit 8 Elementen (`n=8,t=0..7,f=0..7`) *For the sequence with 8 elements*

```
q(t) = {q0 q1 q2 q3 q4 q5 q6 q7}
     = {a0 b0 a1 b1 a2 b2 a3 b3}
     = {a0 0  a1 0  a2 0  a3 0 } + {0  b0 0  b1 0  b2 0  b3}
```

berechnet sich die DHT mit *the DHT is calculated with*

```
Ha0(f) = {A0 A1 A2 A3 | A0 A1 A2 A3}
Hb0(f) = {B0 B1 B2 B3 | B0 B1 B2 B3}
```

und *and*

```
{A0 A1 A2 A3} = Ha(f')
{B0 B1 B2 B3} = Hb(f')
```

zu *to*

```
H(f) = {H0 H1 H2 H3 H4 H5 H6 H7}
```

mit *with*

```
H0 = A0 + B0×cos(  0°) + B0×sin(  0°)
```

```
H1 = A1 + B1×cos( 45°) + B3×sin( 45°)
H2 = A2 + B2×cos( 90°) + B2×sin( 90°)
H3 = A3 + B3×cos(135°) + B1×sin(135°)
H4 = A0 + B0×cos(180°) + B0×sin(180°)
H5 = A1 + B1×cos(225°) + B3×sin(225°)
H6 = A2 + B2×cos(270°) + B2×sin(270°)
H7 = A3 + B3×cos(315°) + B1×sin(315°)
```

Dies kann vereinfacht werden zu *This can be simplified to*

```
H0 = A0 + B0×cos(  0°) + B0×sin(  0°) = A0 + B0
H1 = A1 + B1×cos( 45°) + B3×sin( 45°) = A1 + B1×cos(45°) + B3×sin(45°)
H2 = A2 + B2×cos( 90°) + B2×sin( 90°) = A2 + B2
H3 = A3 + B3×cos(135°) + B1×sin(135°) = A3 - B3×cos(45°) + B1×sin(45°)
H4 = A0 - B0×cos(  0°) - B0×sin(  0°) = A0 - B0
H5 = A1 - B1×cos( 45°) - B3×sin( 45°) = A1 - B1×cos(45°) - B3×sin(45°)
H6 = A2 - B2×cos( 90°) - B2×sin( 90°) = A2 - B2
H7 = A3 - B3×cos(135°) - B1×sin(135°) = A3 + B3×cos(45°) - B1×sin(45°)
```

Mit *With*

```
C1 = B1×cos(45°) + B3×sin(45°)
C3 = B1×sin(45°) - B3×cos(45°)
```

ergibt sich *is*

```
H0 = A0 + B0      (Cog0a)(Cog0a)(Cog0a)
H1 = A1 + C1      (Cog0b)(Cog1b)(Cog1b)
H2 = A2 + B2      (Cog0a)(Cog0a)(Cog0c)
H3 = A3 + C3      (Cog0b)(Cog1b)(Cog1d)
H4 = A0 - B0      (Cog0a)(Cog0a)(Cog0e)
H5 = A1 - C1      (Cog0b)(Cog1b)(Cog1d)
H6 = A2 - B2      (Cog0a)(Cog0a)(Cog0c)
H7 = A3 - C3      (Cog0b)(Cog1b)(Cog1b)
```

Die Größen C1 und C3 können gleichzeitig mittels des Cordic-z0-Verfahrens auf g Stellen genau berechnet werden. Das Cordic-z0-Verfahren bestimmt aus den Eingangsgrößen *The values of C1 and C3 can be calculated simultaneously using the CORDIC-z0-algorithm. The CORDIC-z0-algorithm calculates from the input variables*

```
x,y,z
```

in g+1 Iterationsschritten die Größen *the values*

```
x × cos(z) - y × sin(z)
x × sin(z) + y × cos(z)
```

Mit *With*

```
x =  B1
y = -B3
z =  45°
```

liefert das Cordic-z0-Verfahren die benötigten Werte für C1 und C3. *the CORDIC-z0-algorithm calculates the required values for C1 and C3.*

Für die Folge mit 16 Elementen (`n=16,t=0..15,f=0..15`) ergibt sich als Endergebnis *For the sequence with 16 elements the final result is*

```
H0 = A0 + B0×cos(0°)     + B0×sin(0°)     = A0 + B0
H1 = A1 + B1×cos(22,5°)  + B7×sin(22,5°)
H2 = A2 + B2×cos(45°)    + B6×sin(45°)
H3 = A3 + B3×cos(67,5°)  + B5×sin(67,5°)
H4 = A4 + B4×cos(90°)    + B4×sin(90°)    = A4 + B4
H5 = A5 + B5×cos(112,5°) + B3×sin(112,5°)
H6 = A6 + B6×cos(135°)   + B2×sin(135°)
H7 = A7 + B7×cos(157,5°) + B1×sin(157,5°)
H8 = A0 + B0×cos(180°)   + B0×sin(180°)   = A0 - B0
H9 = A1 + B1×cos(202,5°) + B7×sin(202,5°)
H10= A2 + B2×cos(225°)   + B6×sin(225°)
H11= A3 + B3×cos(247,5°) + B5×sin(247,5°)
H12= A4 + B4×cos(270°)   + B4×sin(270°)   = A4 - B4
H13= A5 + B5×cos(292,5°) + B3×sin(292,5°)
H14= A6 + B6×cos(315°)   + B2×sin(315°)
H15= A7 + B7×cos(337,5°) + B1×sin(337,5°)
```

```
H0 = A0 + B0×cos(0°)     + B0×sin(0°)     = A0 + B0
H1 = A1 + B1×cos(22,5°)  + B7×sin(22,5°)
H2 = A2 + B2×cos(45°)    + B6×sin(45°)
H3 = A3 + B3×cos(67,5°)  + B5×sin(67,5°)
H4 = A4 + B4×cos(90°)    + B4×sin(90°)    = A4 + B4
H5 = A5 + B5×cos(112,5°) + B3×sin(112,5°)
H6 = A6 + B6×cos(135°)   + B2×sin(135°)
H7 = A7 + B7×cos(157,5°) + B1×sin(157,5°)
H8 = A0 - B0×cos(0°)     - B0×sin(0°)     = A0 - B0
H9 = A1 - B1×cos(22,5°)  - B7×sin(22,5°)
H10= A2 - B2×cos(45°)    - B6×sin(45°)
H11= A3 - B3×cos(67,5°)  - B5×sin(67,5°)
H12= A4 - B4×cos(90°)    - B4×sin(90°)    = A4 - B4
H13= A5 - B5×cos(122,5°) - B3×sin(122,5°)
H14= A6 - B6×cos(135°)   - B2×sin(135°)
H15= A7 - B7×cos(157,5°) - B1×sin(157,5°)
```

```
H0 = A0 + B0×cos(0°)     + B0×sin(0°)     = A0 + B0     (Cog0a)(Cog0a)(Cog0a)(Cog0a)
H1 = A1 + B1×cos(22,5°)  + B7×sin(22,5°)  = A1 + C1     (Cog0b)(Cog1b)(Cog1b)(Cog1b)
H2 = A2 + B2×cos(45°)    + B6×sin(45°)    = A2 + C2     (Cog0c)(Cog0c)(Cog2c)(Cog2c)
H3 = A3 + B3×cos(67,5°)  + B5×sin(67,5°)  = A3 + C3     (Cog0d)(Cog1d)(Cog3d)(Cog3d)
H4 = A4 - B4×cos(90°)    + B4×sin(90°)    = A4 + B4     (Cog0a)(Cog0a)(Cog0a)(Cog0e)
H5 = A5 - B5×cos(67,5°)  + B3×sin(67,5°)  = A5 + C5     (Cog0d)(Cog1d)(Cog3d)(Cog1f)
H6 = A6 - B6×cos(45°)    + B2×sin(45°)    = A6 + C6     (Cog0c)(Cog0c)(Cog2c)(Cog2g)
H7 = A7 - B7×cos(22,5°)  + B1×sin(22,5°)  = A7 + C7     (Cog0b)(Cog1b)(Cog1b)(Cog3h)
H8 = A0 - B0×cos(0°)     - B0×sin(0°)     = A0 - B0     (Cog0a)(Cog0a)(Cog0a)(Cog0i)
H9 = A1 - B1×cos(22,5°)  - B7×sin(22,5°)  = A1 - C1     (Cog0b)(Cog1b)(Cog1b)(Cog3h)
H10= A2 - B2×cos(45°)    - B6×sin(45°)    = A2 - C2     (Cog0c)(Cog0c)(Cog2c)(Cog2g)
H11= A3 - B3×cos(67,5°)  - B5×sin(67,5°)  = A3 - C3     (Cog0d)(Cog1d)(Cog3d)(Cog1f)
H12= A4 + B4×cos(90°)    - B4×sin(90°)    = A4 - B4     (Cog0a)(Cog0a)(Cog0a)(Cog0e)
H13= A5 + B5×cos(67,5°)  - B3×sin(67,5°)  = A5 - C5     (Cog0d)(Cog1d)(Cog3d)(Cog3d)
H14= A6 + B6×cos(45°)    - B2×sin(45°)    = A6 - C6     (Cog0c)(Cog0c)(Cog2c)(Cog2c)
```

```
H15= A7 + B7×cos(22,5°)  - B1×sin(22,5°)  = A7 - C7      (Cog0b)(Cog1b)(Cog1b)(Cog1b)
```

Schnelle Berechnung der DHT (Algorithmus) *Fast calculation of DHT (Algorithm)*

Mit dem oben Aufgeführten kann nun ein Algorithmus zur Berechnung der DHT und damit der DFT formuliert werden. Die DHT wird mittels der rekursiven Funktion HT und der einfachen Funktion HTab berechnet. Sie arbeiten auf dem Eingabedatenfeld q[0..N-1] und dem Ausgabedatenfeld H[0..N-1]. Die Funktion HT steuert die Hartley-Transformtation, die Funktion HTab realisiert die Transformations-operationen gemäß der Theoreme `[Gl.04]`, `[Gl.05]` und `[Gl.06]`. *Now with the facts mentioned above, an algorithm to calculate the DFT and thereby the DHT can be formulated. The DHT is calculated by means of the recursive function HT and the simple function HTab. They operate on the input data field q[0..N-1] and the output data field H[0..N-1]. The HT function controls the Hartley transformtation, the HTab function implements the transformation operations according to the theorems `[Gl.04]`, `[Gl.05]` and `[Gl.06]`.*

Für die rekursive Berechnung der DHT müssen aus der Wertefolge *For the recursive computation of the DHT it is necessary to extract from the data sequences*

```
q(t) = {q0 q1 q2 q3 q4 q5 q6 q7 .. q(n-2) q(n-1)     }
     = {a0 b0 a1 b1 a2 b2 a3 b3 .. a(n/2-1) b(n/2-1)}
```

die beiden Folgen *the two data sequences by simple demixing*

```
qa(t) = {a0 a1 a2 a3 .. a(n/2-1)}
qb(t) = {b0 b1 b2 b3 .. b(n/2-1)}
```

durch einfaches Entmischen extrahiert werden. Dies kann für alle während der Rekursionen durchzuführenden Entmischvorgänge in einem Schritt durch Umindizieren der Elemente der Wertefolge q geschehen, sobald auf diese zugegriffen wird. Dazu wird aus dem binär dargestellten Index *This can be done in one step*

```
i = im..i3i2i1i0
```

durch Bitumkehr der Index *by bit reversal*

```
j = i0i1i2i3..im
```

gebildet.

Die Transformation wird mit **HT**`(0,N)` gestartet. *The transformation starts with HT(0,N).*

```
function HT(o,n): H[o..o+n-1]
  if n = 1 then
    H[o+0]:= q[Bitumkehr(o+0)]
  else if n = 2 then
    H[o+0]:= q[Bitumkehr(0+0)] + q[Bitumkehr(o+2)]
    H[o+1]:= q[Bitumkehr(o+0)] - q[Bitumkehr(o+2)]
  else if n = 4 then
    H[o+0]:= q[Bitumkehr(o+0)] + q[Bitumkehr(o+2)] + q[Bitumkehr(o+1)] + q[Bitumkehr(o+3)]
    H[o+1]:= q[Bitumkehr(o+0)] - q[Bitumkehr(o+2)] + q[Bitumkehr(o+1)] - q[Bitumkehr(o+3)]
    H[o+2]:= q[Bitumkehr(o+0)] + q[Bitumkehr(o+1)] - q[Bitumkehr(o+1)] - q[Bitumkehr(o+3)]
    H[o+3]:= q[Bitumkehr(o+0)] - q[Bitumkehr(o+1)] - q[Bitumkehr(o+1)] + q[Bitumkehr(o+3)]
  else
    A:= HT(o,n/2)
    B:= HT(o+n/2,n/2)
    H:= HTab(A,B,o,n/2,360°/n)
```

```
  end
end
function HTab(A,B,o,n,k): H[o..o+2n-1]
  H[o]   := A[0] + B[0]
  H[o+n]:= A[0] - B[0]
  for i:= 1 to n/2-1 do
    x:=  B[i]
    y:= -B[n-i]
    z:=  k*i
    Cordic-z0(x,y,z): C[i],C[n-i]
    H[o+i]     := A[i]   + C[i]
    H[o+n+i]   := A[i]   - C[i]
    H[o+n-i]   := A[n-i] + C[n-i]
    H[o+n+n-i]:= A[n-i] - C[n-i]
  end
  H[o+n/2]   := A[n/2] + B[n/2]
  H[o+n+n/2]:= A[n/2] - B[n/2]
end
```

Das Spektrum der DFT wird aus der DHT mittels der Funktion HTsp berechnet. Diese arbeitet auf dem Eingabefeld H[0..N-1] und dem Ausgabefeld F[0..N-1] und realisiert `[Gl.06]`. Die Umwandlung der komplexen Fourierelemente in Betrag und Phase erfolgt mit dem Cordic-y0-Verfahren auf g Stellen genau. *The spectrum of the DFT is calculated from the DHT using the HTsp function. This operates on the input field H[0..N-1] and the output field F[0..N-1] and realize [Gl.06]. The conversion of the complex Fourier elements in magnitude and phase takes place with the CORDIC-y0-method to g-digit precision.*

Das Cordic-y0-Verfahren bestimmt aus den Eingangsgrößen *The CORDIC-y0-method determines from the input variables*

```
x,y,z
```

in g+1 Iterationsschritten die Größen *in g+1 iteration steps*

```
√(x²+y²)
arctan(y/x)
```

Mit *With*

```
x = Realteil(F) real part
y = Imaginärteil(F) imaginary part
z = 0
```

liefert das Cordic-y0-Verfahren Betrag und Phase eines komplexen Fourierelements. *the CORDIC-y0-method delivers the magnitude and phase of a complex Fourier element.*

```
function HTsp(H[0..n-1],n): F[0..n-1]
  Re:= H[0]
  Im:= 0
  F[0].Be:= abs(Re)
  F[0].Ph:= if Re < 0 then 180° else 0°
  for i:= 1 to n/2-1 do
    Re:= (H[i] + H[n-i]) / 2
    Im:= (H[i] - H[n-i]) / 2
    Cordic-y0(Re,Im,0): Be,Ph
    F[i].Be   :=  Be
    F[i].Ph   :=  Ph
    F[n-i].Be:=  Be
```

```
    F[n-i].Ph:= -Ph
  end
  Re:= H[n/2]
  Im:= 0
  F[n/2].Be:= Be
  F[n/2].Ph:= if Re < 0 then 180° else 0°
end
```

Steht ein Cog zur Verfügung, wird die DFT mittels *If there is one cog available the DFT is computed through*

```
H     := Cog1( HT(0,N) )
F     := Cog1( HTsp(H,N) )
```

bei zwei Cogs mittels *with two cogs through*

```
H_1   := Cog1( HT(0,N/2) )
H_2   := Cog2( HT(N/2,N/2) )
H     := Cog12( HTab(H_1,H_2,N/2) )
F     := Cog12( HTsp(H,N) )
```

und bei vier Cogs mittels *and with four cogs through*

```
H_1   := Cog1( HT(0,N/4) )
H_2   := Cog2( HT(N/4,N/4) )
H_3   := Cog3( HT(N/2,N/4) )
H_4   := Cog4( HT(N×3/4,N/4) )
H_12  := Cog1234( HTab(H_1,H_2,N/4) )
H_34  := Cog1234( HTab(H_3,H_4,N/4) )
H     := Cog1234( HTab(H_12,H_34,N/2) )
F     := Cog1234( HTsp(H,N) )
```

berechnet. Hierbei bedeutet Cog1(), daß die Funktion von Cog1 bearbeitet wird, Cog12(), daß die Funktion von Cog1 und Cog2 gemeinsam bearbeitet wird, und Cog1234(), daß die Funktion von Cog1, Cog2, Cog3 und Cog4 gemeinsam bearbeitet wird. *Herein Cog1() has the meaning that the function is processed by Cog1, Cog12() that the function is processed by Cog1 together with Cog2, and Cog1234() that the function is processed by Cog1 togehter with Cog2, Cog3 and Cog4.*

Bedienanleitung *Operating instructions*

- Das Objekt **Hartley.spin** transformiert eine zeitliche Meßdatenfolge Q[i] in den Frequenzbereich und erzeugt dabei die diskrete Hartley-Transformierte H[i]. Diese kann mittels der entsprechenden Prozeduren als Fourier-Spektrum (**Sp**), Fourier-Transformierte (**Fo**) oder Hartley-Transformierte (**Ha**) dargestellt werden. *The object Hartley.spin transforms a time data sequence Q[i] in the frequency domain and generates the discrete Hartley transform of H[i]. This can by means of the corresponding procedures be represented as Fourier spectrum (Sp), Fourier transform (Fo) or Hartley transform (Ha).*

  Bei der Interpretation dieser Darstellungen ist zu beachten, daß zum einen die Q[i] nur Approximationen der zugrunde liegenden Signale sind, und zum anderen, daß die DHT bzw. die DFT nur Approximationen der Hartley- bzw. Fourier-Transformation sind! *For the interpretation of* *these images it is to be noted that the Q[i] are only approximations of the underlying signal, and that the DHT and the DFT are only approximations of the Hartley or Fourier transform!*

- Folgende Vorbereitungen sind durchzuführen: Einrichten des Long-Arrays **H**[NH] für die Hartley-Transformierte. **NH** muß dabei eine 2er-Potenz sein. Einrichten des zyklischen Word-Arrays **Q**[NQ] für die Meßdatenfolge. **NQ** muß dabei mindestens so groß sein wie NH. Einrichten des Zeigers **s**. Dieser zeigt auf das Element Q[s], ab der die Transformation der Länge NH beginnen soll. *The following preparations are to be performed: Set Long-array H[NH] for the Hartley transform. NH shall be a power of 2. Setting up the cyclic word arrays Q[NQ] for the data sequence. NQ must be at least as large as NH. Set the pointer p. It points to the element Q[s] from which the transform of length NH begins.*

  Bestimmen eines freien Ein-/Ausgabeport für die Triggerung und Synchronisation der Hartley-Transformation. Über diesen Triggerpin **Tp** wird die Transformation soft- oder hardwaremäßig gestartet und angezeigt, ob sie aktiv ist. Bestimmen der Anzahl **NA** der niederwertigsten Bits, die bei der Spektrum-Darstellung ausgeblendet werden können, um eine vernünftige Darstellung der Phasen zu bekommen. Bestimmen der kleinsten Anzahl **NB** der Bits (ohne Vorzeichen), die notwendig sind, um die Meßwerte in der Folge Q[i] darstellen zu können. Bestimmen der Anzahl **NC** der zu verwendenden Cogs. Bestimmen der Darstellung **DH** der Hartley-Transformierten H[i] mittels der Konstanten Sp, Fo, Ha. *Determining a free input/output port for triggering and synchronization of the Hartley transformation. About this Triggerpin Tp the transformation is started by software or hardware and it is indicated whether the transformation is active. Determining the number NA of the least significant bits that can be hidden in the spectrum display to get a reasonable representation of the phases. Determining the smallest number NB of bits (unsigned) that are necessary to represent the values in the sequence Q[i]. Determining the number NC of the cogs to be used. Determining the representation DH of the Hartley transform H[i] using the constants Sp, Fo, Ha.*

  Erzeugen der Initialisierungsvariablen **ard_H, ard_Q, adr_s, val_cfg** aus den Werten für H, NH, Q, NQ, s, Tp, NA, NB, NC, DH und Aufruf der Prozedur **Hartley.Init**. Vor einem weiteren Aufruf von Hartley.Init muß die Prozedur **Hartley.Stop** aufgerufen werden. *Generating the initialization variables ard_H, ard_Q, adr_s, val_cfg from the values for H, NH, Q, NQ, s, Tp, NA, NB, NC, DH. Then call the procedure Hartley.Init. Before another call to the procedure Hartley.Init the procedure Hartley.Stop must be called.*

  Schreiben des zu transformierende Meßdatensatzes in den zyklischen Eingabepuffer Q ab der Startadresse s. *Write the data sequence to be transformed into the cyclic input buffer Q from the start address s.*

  Aufrufen der Prozedur **Hartley.Trig**. Hierdurch wird am Triggerpin Tp ein Impuls erzeugt, der die Transformation startet. Parallel zur Ausführung der Transformation kann der gerade nicht benötigte Teil des zyklischen Meßdatenfeldes Q[i] mit neuen Daten gefüllt werden. *Call the procedure Hartley.Trig. By this a pulse is generated on Triggerpin Tp which starts the* *transformation. In parallel to the execution of the transformation, the at this moment unnecessary portion of the cyclic data field Q[i] can be filled with new data.*

Gegebenenfalls Aufrufen der Prozedur **Hartley.Wait**. Falls eine Transformation gerade aktiv ist, wird solange gewartet, bis über den Triggerpin Tp angezeigt wird, daß die Transformation fertig ist. *If necessary, call the procedure Hartley.Wait. If a transformation is active, it is waited until the Triggerpin Tp indicates that the transformation is complete.*

Gegebenenfalls Aufrufen der Prozedur **Hartley.Sync**. Hierdurch wird erreicht, daß die weitere Verarbeitung unmittelbar nach dem Ende einer Transformation fortgeführt wird. *If necessary, call the procedure Hartley.Sync. Hereby is achieved that the further processing is continued immediately after the end of a transformation.*

Steht die Hartley-Transformierte des Meßdatensatzes Q im Ausgabefeld H zur Verfügung, kann sie über die Funktionen **Hartley.Be(i), Hartley.Ph(i), Hartley.Re(i), Hartley.Im(i)** und **Hartley.We(i)** abgegriffen werden. *If the Hartley transform of the data sequence Q is available in the output field H, the values can be read by the functions Hartley.Be(i), Hartley.Ph(i), Hartley.Re(i), Hartley.Im(i), and Hartley.We(i).*

```
• Initialisierung:      Hartley.Init( ard_H , ard_Q , adr_s , val_cfg ) Initialization
  Arraydeskriptor:      ard_H   = @H + NH<<16 Array descriptor
  Arraydeskriptor:      ard_Q   = @Q + NQ<<16 Array descriptor
  Variablenadresse:     adr_s   = @s Variable address
  Konfigurationswert:   val_cfg = Tp<<16 + NA<<12 + NB<<8 + NC<<4 + DH Configuration value

• Starten:              Hartley.Trig Start

• Abwarten:             Hartley.Wait Wait

• Synchronisieren:      Hartley.Sync Synchronize

• Anhalten:             Hartley.Stop Stop

• NH:
  Wertebereich = 8,16,32..2048.. Value range

• long H [NH]:
  Sp-Wertebereich = Value range
     Betrag:         ±i(NB-1)..i0.f1..f(31-NB) = 0..2^NB*(1-2^-31) Magnitude
     Phase:          ±i7..i0.f1..f7 = -180.00° <= Phase <= 180.00° Phase
  Fo-Wertebereich = Value range
     Realwert:       ±i(NB-1)..i0.f1..f(31-NB) = -2^NB..2^NB*(1-2^-31) Real value
     Imaginärwert:   ±i(NB-1)..i0.f1..f(31-NB) = -2^NB..2^NB*(1-2^-31) Imaginary value
  Ha-Wertebereich = Value range
     Hartleywert:    ±i(NB-1)..i0.f1..f(31-NB) = -2^NB..2^NB*(1-2^-31) Hartley value

  Sp-Zugriff (Fourier-Spektrum) = Access
     Betrag:        Hartley.Be(i) Magnitude
     Phase:         Hartley.Ph(i) Phase
  Fo-Zugriff (Fourier-Transformierte) = Access
     Realwert:      Hartley.Re(i) Real value
     Imaginärwert:  Hartley.Im(i) Imaginary value
```

Ha-Zugriff (Hartley-Transformierte) = *Access*

Hartleywert: **Hartley.We(i)** *Hartley value*

- **NQ:**

  Wertebereich ≥ NH *Value range*

- **word Q** [NQ]:

  Wertebereich = -2^NB..2^NB-1 oder *or* 0..2^NB-1 *Value range*

- **word s:**

  Wertebereich = 0..NQ-1 *Value range*

- **Tp:**

  Wertebereich = 0..31 *Value range*

- **NA:**

  Wertebereich = 0..15 Bits *Value range*

- **NB:**

  Wertebereich = 0(=16),1..15 Bits *Value range*

- **NC:**

  Wertebereich = 1,2,4 Cogs *Value range*

- **DH:**

  Wertebereich = 0,1,2 = Sp,Fo,Ha *Value range*

Beispiele *Examples*

- In den folgenden Beispielen liegt der Triggerpin auf Port 27 und die TV-Pins liegen auf den Ports 23..20. In den graphischen Darstellungen sind die Datenfolgen blau, die Beträge, Realwerte und Hartleywerte grün, die Phasen und Imaginärwerte rot. *In the following examples the Triggerpin is on port 27 and the TV pins are on ports 23..20. These assignments can be changed in the Demo and Display objects themselves. In the displayed graphs, the data sequences are blue, the magnitude, real and Hartley values are green, the phases and imaginary values are red.*

- Hartley_Demo#1.spin:

  Beispiel aus Brackwell S. 59, Tab. 4.1

  Hartley-Transformierte der Folge [20 15 6 1 0 0 0 0 0 0 0 0 0 1 6 15]

i	f	H
0	20	4.000
1	15	3.560
2	6	2.487
3	1	1.322
4	0	0.500
5	0	0.118
6	0	0.013
7	0	0.000

8	0	0.000
9	0	0.000
10	0	0.013
11	0	0.118
12	0	0.500
13	1	1.322
14	6	2.487
15	5	3.560

- `Hartley_Demo#2.spin:`

  Beispiel aus Brackwell S. 62, Tab. 4.2, f1

  Hartley-Transformierte der Folge `[0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0]`

i	f1	DHT1
0	0	1.000
1	0	0.250
2	0	-0.854
3	0	0.374
4	1	0.000
5	1	0.250
6	1	-0.354
7	1	0.050
8	0	0.000
9	0	0.250
10	0	-0.146
11	0	-0.167
12	0	0.000
13	0	0.250
14	0	-0.354
15	0	-1.257

- `Hartley_Demo#3.spin:`

  Beispiel aus Brackwell S. 62, Tab. 4.2, f2

  Hartley-Transformierte der Folge `[1 1` **`1`** `0 0 0 0 0 0 0 0 0 0 0 1 1]` und nicht der Folge *and not*
  `[1 1` **`0`** `0 0 0 0 0 0 0 0 0 0 0 1 1]`

i	f2	DHT2
0	1	1.0000
1	1	0.8524
2	1	0.4828
3	0	0.0702

4	0	-0.2000
5	0	-0.2359
6	0	-0.0828
7	0	0.1133
8	0	0.2000
9	0	0.1133
10	0	-0.0828
11	0	-0.2359
12	0	-0.2000
13	0	0.0702
14	1	0.4828
15	1	0.8524

- `Hartley_Demo#4.spin:`

  Beispiel aus Brackwell S. 62, Tab. 4.2, DHT2

  Hartley-Transformierte der Folge `[1024 873 495 72 -205 -242 -85 116 205 116 -85 -242 -205 72 495 873]`

  In diesem Beispiel wird die **inverse** DHT angewendet, um die Folge f2 zu bestimmen, die mit der in der Tabelle angegebenen DHT2 korrespondiert. Hierzu wird die DHT2 mit 1024 multipliziert und transformiert. *In this example, the inverse DHT is employed to determine the data sequence f2, corresponding to the DHT2 indicated in the table. For this, the DHT2 is multiplied by 1024 and transformed.*

i	DHT2	f2	=
0	1024	204.81	1
1	873	204.89	1
2	495	204.88	1
3	72	-0.13	0
4	-205	-0.06	0
5	-242	-0.03	0
6	-85	-0.00	0
7	116	0.01	0
8	205	0.06	0
9	116	0.01	0
10	-85	-0.00	0
11	-242	-0.03	0
12	-205	-0.06	0
13	72	-0.13	0
14	495	204.88	1
15	873	204.90	1

- `Hartley_Demo#5.spin:`

  Beispiel aus Brackwell S. 235

  Hartley-Transformierte der Folge `[1 2 3 4 5 6 7 8 9 .. 256]`

  Durch Setzen der Spalte D in HTcfg von 0 auf 1 oder 2 kann die Darstellung geändert werden. *By setting the column D in HTcfg from 0 to 1 or 2, the display can be changed.*

i	q[i]	H	We'S	Re	Im	Im'S	Be	Ph[°]	Ph[°]'C	Ph[°]'S
0	1	128.500	128.500	128.500	0.000	0.000	128.500	0.000	0.000	0.000
1	2	-41.242	-41.241	-0.500	-40.742	-40.741	40.745	-90.703	-90.703	-90.703
2	3	-20.868	-20.868	-0.500	-20.368	-20.368	20.374	-91.406	-91.406	-91.406
3	4	-14.075	-14.075	-0.500	-13.575	-13.575	13.584	-92.109	-92.109	-92.109
4	5	-10.678	-10.678	-0.500	-10.178	-10.178	10.190	-92.813	-92.812	-92.812
5	6	-8.639	-8.638	-0.500	-8.139	-8.138	8.154	-93.516	-93.517	-93.515
6	7	-7.278	-7.278	-0.500	-6.778	-6.778	6.797	-94.219	-94.218	-94.220
7	8	-6.306	-6.306	-0.500	-5.806	-5.806	5.828	-94.922	-94.922	-94.922
8	9	-5.577	-5.577	-0.500	-5.077	-5.077	5.101	-95.625	-95.625	-95.625
9	10	-5.009	-5.009	-0.500	-4.509	-4.509	4.536	-96.328	-96.330	-96.328
.	.	.	.	.	.	.	.	.	.	.

- `Hartley_Demo#Sin.spin:`

  Hartley-Transformierte einer 1 Hz Sinusfunktion mit 256 Abtastpunkten *Sine function*

- `Hartley_Demo#Cos.spin:`

  Hartley-Transformierte einer 1 Hz Cosinusfunktion mit 256 Abtastpunkten *Cosine function*

- `Hartley_Demo#Rec.spin:`

  Hartley-Transformierte einer 1 Hz Rechteckfunktion mit 256 Abtastpunkten *Rectangular function*

  Die über die Abtastpunkte definierte Funktion stellt in Wirklichkeit keine Rechteckfunktion sondern eine Trapezfunktion dar. Deshalb unterscheidet sich das Spektrum von dem einer idealen Rechteckfunktion, insbesondere nimmt die Phase, frequenzabhängig von 90° bis 0° ab, anstatt konstant 90° zu sein. *The function defined through the sampling points is in fact not a square function but a trapezoidal function. Therefore, the spectrum differs from the ideal square wave function, and in particular the phase decreases from 90° to 0° instead of being constant 90°.*

- `Hartley_Demo#Tri.spin:`

  Hartley-Transformierte einer 1 Hz Dreieckfunktion mit 256 Abtastpunkten *Triangle function*

- `Hartley_Demo#Saw.spin:`

  Hartley-Transformierte einer 1 Hz Sägezahnfunktion mit 256 Abtastpunkten *Sawtooth function*

- `Hartley_Demo#Ewg.spin:`

  Hartley-Transformierte einer 1 Hz Einweggleichrichtung mit 256 Abtastpunkten *Half-wave rectification*

- `Hartley_Demo#Zwg.spin:`

  Hartley-Transformierte einer 1 Hz Zweiweggleichrichtung mit 256 Abtastpunkten *Full-wave rectification*

Rechenzeiten (80 MHz) *Computation times*

NH	1 Cog [ms] Ha,Fo	1 Cog [ms] Sp	2 Cogs [ms] Ha,Fo	2 Cogs [ms] Sp	4 Cogs [ms] Ha,Fo	4 Cogs [ms] Sp
8	0.063	0.082	nicht möglich	nicht möglich	nicht möglich	nicht möglich
16	0.117	0.192	0.076	0.119	nicht möglich	nicht möglich
32	0.302	0.462	0.168	0.253	0.103	0.145
64	0.767	1.098	0.401	0.571	0.219	0.305
128	1.897	2.568	0.965	1.307	0.502	0.673
256	4.547	5.899	2.291	2.972	1.164	1.506
512	10.631	13.344	5.332	6.694	2.686	3.366
1024	24.364	29.799	12.200	14.921	6.119	7.480
2048	54.965	65.842	27.499	32.944	13.769	16.492

Programmtechnische Erläuterungen *Notes to the programs*

- ### Sinustabelle *Sine table (tested but not used in the programm)*

  Die Propeller-Sinustabelle ist auf `0.FFFF` normiert. Dies entspricht einer Multiplikation der Sinuswerte mit `(1-2^-16)`. Um die Sinuswerte wieder auf `1.0000` zu normieren, müssen diese durch `(1-2^-16)` dividiert werden. Diese Division kann sehr gut durch eine Multiplikation mit `(1+2^-16)` angenähert werden. *The propeller sine table is scaled to `0.FFFF`. This corresponds to a multiplication of the sine values by `(1-2^-16)`. To standardize the sine values back to `1.0000`, they must be divided by `(1-2^-16)`. This division can be very well approximated by multiplying with `(1+2^-16)`.*

- ### Arcustangenstabelle *Arcus tangent table*

  Das Cordic-Verfahren verwendet eine spezielle Arcustangenstabelle. Damit der Index in diese Tabelle gleichzeitig als Index für die Shiftoperationen verwendet werden kann, muß der Tabellenanfang auf einer durch 32 teilbaren Adresse liegen. Die Verwendung des Cordic-Verfahren ist trotz der notwendigen Skalierungen effizienter als die Verwendung der Propeller-Sinustabelle. *The CORDIC technique uses a special arcus tangent table. In order to use the index into this table simultaneously as an index for the shift operations, the table top must be on a address divisible by 32. The use of the cordic method is more efficient than the use of the propeller sine table despite the necessary scaling.*

- ### Rekursion *Recursion*

  Der für die Hartley-Transformation verwendete rekursive Algorithmus läßt sich effizient auf dem Propeller implementieren. Als Return-Stack reicht eine Long-Variable, da keine Rücksprungadresse, sondern nur die Information, ob beim rekursiven Abstieg der linke oder der rechte Zweig gewählt wurde, abgespeichert wird. Ein Parameter-Stack ist nicht erforderlich, da sich beim rekursiven Abstieg die neuen Parameter aus den alten Paramertern eineindeutig **berechnen** lassen. Daher können beim rekursiven Aufstieg die alten Parameter **neu** berechnet und damit wiederhergestellt werden. *The recursive algorithm used for the Hartley transform can be implemented efficiently on the propeller. For the return stack a long variable is sufficient, since no return address is stored, but only the information whether the left or the right branch was selected on recursive descent. A parameter stack is not required, since the new parameters can be calculated from the old ones on recursive descent. Therefore, on recursive ascent the old parameters can be restored by recalculating.*

- ### Parallelisierung *Parallelization*

Der rekursive Algorithmus für die Hartley-Transformation spannt einen binären Baum auf, dessen Teilbäume unabhängig voneinander berechnet werden können. Die dabei entstehenden Teilergebnisse können dann mittels der Funktion HTab zusammengeführt werden. Die Funktion HTab ist so ausgelegt, daß sie sowohl von einem als auch von zwei oder vier Cogs gleichzeitig berechnet werden kann. Stehen z. B. zwei Cogs zur Verfügung, wird die Datenfolge halbiert und jedem Cog eine Hälfte zugewiesen. Da die Rechenzeiten für die Funktionen HT und HTab von der Datenfolge unabhängig sind, werden die beiden Cogs gleichzeitig fertig. Das Zusammenführen der Ergebnisse teilen sich die beiden Cogs, indem jeder Cog eine Hälfte der in der Funktion HTab durchzuführenden Berechnungen ausführt. Die Rechenzeiten der beiden Cogs sind aufgrund der Programmlogik der Funktion HTab unterschiedlich, es ist aber durch eine inhärente Eigenschaft der Programmlogik sichergestellt, daß bei der nachfolgende Berechnung des Spektrums durch die Funktion HTsp alle benötigten Werte zur Verfügung stehen. Entsprechendes gilt bei vier Cogs auch für die Funktion HTsp selber. *The recursive algorithm for the Hartley transform spans a binary tree whose subtrees can be computed independently. The resulting partial results may then be combined by means of the function HTab. HTab is designed such that it can be calculated by one or two or four cogs simultaneously. If for example two cogs are available, the data sequence is halved, and each half is assigned to one of the two cogs. Since the computation times for the functions HT and HTab are independent from the data sequence, the two cogs finish simultaneously. The merging of the computed results is carried out simultaneously by the two cogs, each cog by performing a half of the calculations carried out in the HTab function. The computation times of the two cogs are different due to the logic of the HTab function. However, it is ensured by an inherent feature of the program logic, that when the subsequent calculation of the spectrum by the function HTsp will be done all necessary values are available. The same applies to four cogs for the HTsp function itself.*

Literatur

Bracewell, Ronald H.
Schnelle Hartley-Transformation
Eine reellwertige Alternative zur FFT
München: Oldenbourg, 1990