BW6 over BLS12-381

Authors: Youssef El Housni (ConsenSys, GRACE) and Aurore Guillevic (inria)

Following the recent work on FFT over non-smooth fields (ECFFT

), it might be viable to instantiate a SNARK with an elliptic curve of non-smooth subgroup order. Particularly, one layer SNARK composition can be achieved with less constraints for the choice of curves. In this note we propose a 2-chain based on the widely used BL12-381 curve. The outer curve is a BW6 with a subgroup order 2-adicity equal to one.

ECFFT

For smooth finite fields $\mathbb{F_q}$

(i.e., when $q-1$

factors into small primes) the Fast Fourier Transform (FFT

) leads to the fastest known algebraic algorithms for many basic polynomial operations, such as multiplication, division, interpolation and multi-point evaluation. However, the same operations over fields with no smooth order root of unity suffer from an asymptotic slowdown. [In a recent work by Ben-Sasson, Carmon, Kopparty and Levit, called [ECFFT

](https://arxiv.org/abs/2107.08473), the authors proposed a new approach to fast algorithms for polynomial operations over all large finite fields.] The key idea is to replace the group of roots of unity with a set of points $L \subset F$

suitably related to a well-chosen elliptic curve group (the set L

itself is not a group). The key advantage of this approach is that elliptic curve groups can be of any size in the Hasse-Weil interval $[q+1\pm2\sqrt{q}]$

and thus can have subgroups of large, smooth order, which an FFT-like divide and conquer algorithm can exploit.

(From the paper's abstract

)

2-chains

In HG20, the authors proposed a method to construct a fast pairing-friendly elliptic curve over BLS12-377. That is, a curve E_2

with a subgroup order r_2

equal to field size q_1

of BLS12-377. This method can be generalized to any inner curve and especially to BLS12 curves. The choice of BLS12-377 was interesting because $q_1-1 \equiv 0 \mod 2^{46}$

yielding efficient radix-2 FFT needed to generate SNARK proofs over the outer BW6 curve. With the ECFFT

breakthrough, one can swap BLS12-377 for the more widely used BLS12-381 and construct a new BW6 outer curve. This latter curve would have a subgroup order r_2

s.t. $r_2-1 \equiv 0 \mod 2$

, but ECFFT

approach might lead to acceptably efficient polynomial operations over $\mathbb{F}_{r_2}$

.

Inner curve: BLS12-381

BLS12-381 is a pairing-friendly elliptic curve from the family Barreto-Lynn-Scott with an embedding degree $k=12$

(Construction 6.6 in FST06, case $k\equiv 0 \mod 6$

). It is defined over a 381-bit field $\mathbb{F}_q$

and is defined by the equation $y^2=x^3+4$

. The curve parameters are parametrized by polynomials that are evaluated at the seed $u=-15132376222941642752$

.

Parameter	Polynomial	Value	2-adicity
field size q	$(x-1)^2/3\cdot r(x)+x$	0x1a0111ea397fe69a4b1ba7b6434bacd764774b84f38512bf6730d2a0f6b0f6241eabfffeb153ffffb9feffffffffaaab	1
subgroup order r	x^4-x^2+1	0x73eda753299d7d483339d80809a1d80553bda402fffe5bfeffffffff00000001	32

Besides the sizes of q

, r

and the the low hamming-weight of u

, which makes the pairing-friendly curve efficient and secure at almost 128-bit level, the high 2-adicity of $r-1$

makes it a fit for SNARK applications. However, the low 2-adicity of $q-1$

makes it inefficient to construct an outer curve with subgroup order $r_2=q$

.

Outer curve: BW6-767

Following HG20, we're looking for a curve E_2

with a subgroup order r_2

equal to BLS12-381 q

. The same observations apply: we need q_2

to be less than 768 bits and $D=-3$

. We use the modified Brezing-Weng method with tailored Frobenius trace $t(x)=t_0(x)+h_t \cdot r(x)$

and CM parameter $y(x)=y_0(x)+h_y \cdot r(x)$

(from CM equation $4q=t^2+Dy^2$

). The lifting cofactors h_t

and h_y

must be chosen carefully according to the desired bit lengths and the CM discriminant $-D$

. Given these constraints, we found 4 curves, which boils down basically to two pair of curves (a curve and its twist over the base field $\mathbb{F}_q$

).

Our script, which will be soon open-sourced under MIT license, outputs the following curves:

test_vector_BLS12_381_BW6_768 = [{'u':-0xd201000000010000, 'D':3, 'ht':-4, 'hy':-6, 'b':1, 'pnbits':767, 'rnbits':381, 'px':[31,65,37,115,112,-100,56,29,-191,79,142,-127,31], 'px_denom':9, 'rx':[1,1,0,2,0,-2,1], 'rx_denom':3, 'cx':[52,16,21,41,12,-65,31], 'cx_denom':3, 'yx':[6,7,0,9,3,-13,6], 'yx_denom':3, 'tx':[-4,-1,0,-17,9,5,-4], 'tx_denom':3, 'betax':[72,62,-16,258,-86,-152,234,-97,-190,300,-158,31], 'betax_denom':33, 'lambx':[1,-1,0,3,-3,1], 'lambx_denom':1, 'g2cx':[19,16,21,41,12,-65,31], 'g2cx_denom':3, 'label':"BLS12_BW6_767"}, {'u':-0xd201000000010000, 'D':3, 'ht':5, 'hy':9, 'b':9, 'pnbits':768, 'rnbits':381, 'px':[67,128,64,286,238,-262,209,38,-515,304,241,-262,67], 'px_denom':9, 'rx':[1,1,0,2,0,-2,1], 'rx_denom':3, 'cx':[61,43,21,140,-15,-128,67], 'cx_denom':3, 'yx':[9,8,0,21,-3,-17,9], 'yx_denom':3, 'tx':[5,8,0,1,9,-13,5], 'tx_denom':3, 'betax':[186,107,-43,663,-293,-359,729,-511,-199,570,-329,67], 'betax_denom':48, 'lambx':[1,-1,0,3,-3,1], 'lambx_denom':1, 'g2cx':[109,43,21,140,-15,-128,67], 'g2cx_denom':3, 'label':"BLS12_BW6_768"}, {'u':-0xd201000000010000, 'D':3, 'ht':7, 'hy':5, 'b':3, 'pnbits':767, 'rnbits':381, 'px':[31,65,37,115,112,-100,56,29,-191,79,142,-127,31], 'px_denom':9, 'rx':[1,1,0,2,0,-2,1], 'rx_denom':3, 'cx':[19,16,21,41,12,-65,31], 'cx_denom':3, 'yx':[5,4,0,13,-3,-9,5], 'yx_denom':3, 'tx':[7,10,0,5,9,-17,7], 'tx_denom':3, 'betax':[72,62,-16,258,-86,-152,234,-97,-190,300,-158,31], 'betax_denom':33, 'lambx':[1,-1,0,3,-3,1], 'lambx_denom':1, 'g2cx':[52,16,21,41,12,-65,31], 'g2cx_denom':3, 'label':"BLS12_BW6_767"}, {'u':-0xd201000000010000, 'D':3, 'ht':-11, 'hy':-7, 'b':3, 'pnbits':768, 'rnbits':381, 'px':[67,128,64,286,238,-262,209,38,-515,304,241,-262,67], 'px_denom':9, 'rx':[1,1,0,2,0,-2,1], 'rx_denom':3, 'cx':[109,43,21,140,-15,-128,67], 'cx_denom':3, 'yx':[7,8,0,11,3,-15,7], 'yx_denom':3, 'tx':[-11,-8,0,-31,9,19,-11], 'tx_denom':3, 'betax':[186,107,-43,663,-293,-359,729,-511,-199,570,-329,67], 'betax_denom':48, 'lambx':[1,-1,0,3,-3,1], 'lambx_denom':1, 'g2cx':[61,43,21,140,-15,-128,67], 'g2cx_denom':3, 'label':"BLS12_BW6_768"}]

All these curves benefit from endomorphism-based optimizations as $D=-3$

and from fast Miller loop as the seed u

has low hamming-weight (same as BLS12-381).

That is said, the best curve is the first one: $y^2=x^3+1$

defined over a 767-bit field with $D=-3$

, $h_t=-4$

and $h_y=-6$

.

- The field $\mathbb{F}_q$

can be implemented in twelve 64-bit limbs, with one bit to spare for carries or for compressed y-coordinate flags.

- The cofactors h_t,h_y

give the fastest final exponentiation in the pairing computation as an exponentiation by $h_t^2+3h_y^2$

is needed.

- The coefficient $b=1$

is convenient for checking that a point is on the curve.

- There is a 2-isogeny from a curve with $j \ne 0$

or 1728

, allowing use of the “simplified SWU” method for hashing to an elliptic curve.

The constant $c=3$

is a sextic non-residue in $\mathbb{F}_q$

and so the twisted curve is a M-twist that corresponds to the third output curve: $y^2=x^3+3$

over the same field $\mathbb{F}_q$

with $D=-3$

, $h_t=7$

and $h_y=5$

. The extension fields can be constructed as $\mathbb{F}_{q^3}[u]=\mathbb{F}_q/u^3-3$

and $\mathbb{F}_{q^6}[v]=\mathbb{F}_{q^3}/v^2-u$

.

$\rightarrow$

Below are the polynomial forms of the curves’ parameters:

Curve

subgroup order $r(x)$

(same as BLS12-381 $q(x)$

)

field size $q(x)$

cofactor $c(x)$

Frobenius trace $t(x)$

CM $y(x)$

GLV $\lambda(x)$

GLV $\beta(x)$

$y^2=x^3+1$

$(x-1)^2/3\cdot (x^4-x^2+1)+x$

(31x^{12} - 127x^{11} + 142x^{10} + 79x^9 - 191x^8 + 29x^7 + 56x^6 - 100x^5 + 112x^4 + 115x^3 + 37x^2 + 65x + 31)/9

(93x^6 - 195x^5 + 36x^4 + 123x^3 + 63x^2 + 48x + 156)/9

(-4x^6 + 5x^5 + 9x^4 - 17x^3 - x - 4)/3

(6x^6 - 13x^5 + 3x^4 + 9x^3 + 7x + 6)/3

1-x+3x^3-3x^4+x^5

(72+62x-16x^2+258x^3-86x^4-152x^5+234x^6-97x^7-190x^8+300x^9-158x^{10}+31x^{11})/33

y^2=x^3+3

(twist)

(x-1)^2/3\cdot (x^4-x^2+1)+x

(31x^{12} - 127x^{11} + 142x^{10} + 79x^9 - 191x^8 + 29x^7 + 56x^6 - 100x^5 + 112x^4 + 115x^3 + 37x^2 + 65x + 31)/9

(93x^6 - 195x^5 + 36x^4 + 123x^3 + 63x^2 + 48x + 57)/9

(7x^6 - 17x^5 + 9x^4 + 5x^3 + 10x + 7)/3

(5x^6 - 9x^5 - 3x^4 + 13x^3 + 4x + 5)/3

1-x+3x^3-3x^4+x^5

(72+62x-16x^2+258x^3-86x^4-152x^5+234x^6-97x^7-190x^8+300x^9-158x^{10}+31x^{11})/33

where \lambda(x)

is the endomorphism eigenvalue which when multiplied by a point P=(x_P,y_P)

on the curve acts as (x_P,y_P) \mapsto (\beta(x)\cdot x_P, y_P)

.

\rightarrow

Below are the hexadecimal values of the shared curves' parameters, when evaluated at the seed u=-0151323762229416427 52

:

Curve

subgroup order r

(same as BLS12-381 q

)

field size q

GLV \beta

GLV \lambda

y^2=x^3+1

and y^2=x^3+3

(twist)

0x1a0111ea397fe69a4b1ba7b6434bacd764774b84f38512bf6730d2a0f6b0f6241eabfffeb153ffffb9feffffffffaaab

0x51e2bcf25fa8992238259ea59a063294c36dc4098befce4230f8d18f41e3fc19665e4360b872007d3dd5a1b865cbe8dadc2ce0c034926d18fe0ef8c1c63df7d97cbc118805598e5c31732000974254c83a38b

0x4a7108dc56f65caaa3748933617f8fb542e1d6e9ef88f166a011051b5f65b0b728456c117f8e93508e38b5b4682991ce3dae358b1b36f832f0bb174392eb6c806d9cd8a33550dbe01e63601ff328f27c5a594f

0x1a0111ea397fe699ec02408663d4de85aa0d857d89759ad4897d29650fb85f9b409427eb4f49fffd8bfd00000000aaac

Implementation

A first version of an optimized implementation of the curve arithmetic and the underlying finite fields, alongside optimal ate pairing computation (with fast final exp) was written in Golang under gnark-crypto library (zkTeam @ConsenSys

). It can be found in this fork's branch: gnark-crypto/ecc/bw6-767 at feat/bw6_on_bls12-381 · yelhousni/gnark-crypto · GitHub

\rightarrow

Some more optimizations are WIP:

- [ ] fast cofactor clearing
- [ ] fast subgroups check
- [ ] hash-to-curve

Conclusion

For interoperability of blockchain projects, it would be great to use the same elliptic curve. BLS12-381 is a secure, optimized and widely used pairing-friendly elliptic curve in many patforms. For projects that need one layer proof composition, it might be viable to stick to BLS12-381 and use this BW6-767 alongside ECFFT for composing proofs.

A detailed paper will be published soon that proposes a generic framework for constructing optimized 2-chains for SNARK composition with open-sourced code.