定点化数学运算库的结构

层次一：skv_math_core

两个实数之间的加减乘除，通过宏定义实现定标定点化运算，我们定义了上百个各种情况的宏定义，比如

```
#define MULT16_16_Q11_32(a,b) (SHR(MULT16_16((a),(b)),11))
#define MULT16_16_Q13(a,b) (SHR(MULT16_16((a),(b)),13))
#define MULT16_16_Q14(a,b) (SHR(MULT16_16((a),(b)),14))
#define MULT16_16_Q15(a,b) (SHR(MULT16_16((a),(b)),15))
```

层次二：skv_math_middle

1） 两个复数和复数与实数之间的加减乘除，通过内联函数实现。复数的实部虚部加减乘除计算通过调用**层次一 skv_math_core** 中定义的定标定点运算符。比如

```
static inline Complex Add1(Complex a, Complex b)
static inline Complex Add2(spx_word16_t a, Complex b)
static inline Complex Add3(Complex a, spx_word16_t b)
```

2） 向量的点积，也就是乘加和运算，通过正常的函数实现。复数的实部虚部加减乘除以及实数之间的加减乘除计算通过调用**层次一 skv_math_core** 中定义的定标定点运算符，比如

```
Complex inner_product_complex(Complex *a, Complex *b, spx_uint32_t len);
spx_word16_t inner_product_real(spx_word16_t *a, spx_word16_t *b, spx_uint32_t len);
Complex inner_product_complex_real(Complex *a, spx_word16_t *b, spx_uint32_t len);
Complex inner_product_real_complex(spx_word16_t *a, Complex *b, spx_uint32_t len);
```

3） 向量之间点乘点除点加点减，也就是对应元素的加减乘除，通过正常的函数实现。复数的实部虚部加减乘除以及实数之间的加减乘除计算通过调用**层次一 skv_math_core** 中定义的定标定点运算符，比如

```
Complex* dot_product_complex1(Complex *a, Complex *b, Complex *out, spx_uint32_t len);
Complex* dot_product_complex2(Complex a, Complex *b, Complex *out, spx_uint32_t len);
Complex* dot_product_complex3(Complex *a, Complex b, Complex *out, spx_uint32_t len);
spx_word16_t* dot_product_real1(spx_word16_t *a, spx_word16_t *b, spx_word16_t *out, spx_uint32_t len);
spx_word16_t* dot_product_real2(spx_word16_t a, spx_word16_t *b, spx_word16_t *out, spx_uint32_t len);
spx_word16_t* dot_product_real3(spx_word16_t *a, spx_word16_t b, spx_word16_t *out, spx_uint32_t len);
Complex* dot_product_complex_real1(Complex *a, spx_word16_t *b, Complex *out, spx_uint32_t len);
Complex* dot_product_complex_real2(Complex a, spx_word16_t *b, Complex *out, spx_uint32_t len);
Complex* dot_product_complex_real3(Complex *a, spx_word16_t b, Complex *out, spx_uint32_t len);
Complex* dot_product_real_complex1(spx_word16_t *a, Complex *b, Complex *out, spx_uint32_t len);
Complex* dot_product_real_complex2(spx_word16_t a, Complex *b, Complex *out, spx_uint32_t len);
```

```
Complex* dot_product_real_complex3(spx_word16_t *a, Complex b, Complex *out, spx_uint32_t len);
```

层次三：skv_math

矩阵之间的加减乘除运算，矩阵的所有运算通过调用**层次二：skv_math_middle** 关于向量直接的运算实现。