



shuffles

jonathan garrett

3/24/08

introduction

- we use a standard shuffle naming convention
- very useful, as the name specifies the shuffle (eg. what does **shuf_top** do ?)
- from ICE – very similar scheme from SCEE-ATG

scheme

- characters **A–P** specify bytes **0–15**
- special characters:
 - **0** → **0x00**
 - **X** or **x** → **0xff**
 - **8** → **0x80**
- words use **A–D**
shorts use **A–H**
bytes use **A–P**
- upper case from qw 1 - lower case from qw 2

scheme contd.

x				y				z				w			
03	02	01	00	07	06	05	04	0b	0a	09	08	0f	0e	0d	0c
			A				B				C				D
	B		A		D		C		F		E		H		G
D	C	B	A	H	G	F	E	L	K	J	I	P	O	N	M

examples

- **shuf_ABCd**

- words 0, 1, 2 from qw 1
- word 3 from qw 2

```
0x00, 0x01, 0x02, 0x03,  
0x04, 0x05, 0x06, 0x07,  
0x08, 0x09, 0x0a, 0x0b,  
0x1c, 0x1d, 0x1e, 0x1f
```

examples contd.

- **shuf_AB00CD00ef0088xx**
 - bytes 0, 1 from qw 1
 - fill two bytes with 0x00
 - bytes 2, 3 from qw 1
 - fill two bytes with 0x00
 - bytes 4, 5 from qw 2
 - fill two bytes with 0x00
 - fill two bytes with 0x80
 - fill two bytes with 0xff

```
0x00, 0x01, 0x80, 0x80,  
0x02, 0x03, 0x80, 0x80,  
0x14, 0x15, 0x80, 0x80,  
0xe0, 0xe0, 0xc0, 0xc0
```

examples contd.

- short → float:

```
v01234567 = | s0 s1 | s2 | ... |s7|
v0123     = shufb v01234567, v01234567, shuf_A0B0C0D0
v4567     = shufb v01234567, v01234567, shuf_E0F0G0H0
v0123     = csflt v0123, v0123, ...
v4567     = csflt v4567, v4567, ...
```

- shuffle into high short to avoid sign-extension

- **shuf_A0B0C0D0**

```
0x00, 0x01, 0x80, 0x80,
0x02, 0x03, 0x80, 0x80,
0x04, 0x05, 0x80, 0x80,
0x06, 0x07, 0x80, 0x80
```

- **shuf_E0F0G0H0**

```
0x08, 0x09, 0x80, 0x80,
0x0a, 0x0b, 0x80, 0x80,
0x0c, 0x0d, 0x80, 0x80,
0x0e, 0x0f, 0x80, 0x80
```

helper macros

- here's some C macros which allow declaring shuffles using this convention (originally from ICE, modified by Insomniac, but overall probably not that useful!):

- `shuf_A0B0C0D0 = SHUF(A0B0C0D0);`

or

- `DSHUF(A0B0C0D0);`

helper macros contd.

```
/////////////////////////////////////////////////////////////////
//
// shuffle helpers - note: anything unexpected goes to 0
//
/////////////////////////////////////////////////////////////////
#define SHUFB_ELEM(x_) ( ((x_)>='A' && (x_)<='P') ? (0x00 + (0x01 * ((u32)(x_)-'A'))) : \
                        ((x_)>='a' && (x_)<='p') ? (0x10 + (0x01 * ((u32)(x_)-'a'))) : \
                        ((x_)=='0') ? (0x80) : \
                        ((x_)=='x' || (x_)=='X') ? (0xc0) : \
                        ((x_)=='8') ? (0xe0) : \
                        (0x00) )

#define SHUFH_ELEM(x_) ( ((x_)>='A' && (x_)<='H') ? (0x0001 + (0x0202 * ((u32)(x_)-'A'))) : \
                        ((x_)>='a' && (x_)<='h') ? (0x1011 + (0x0202 * ((u32)(x_)-'a'))) : \
                        ((x_)=='0') ? 0x8080 : \
                        ((x_)=='x' || (x_)=='X') ? 0xc0c0 : \
                        ((x_)=='8') ? 0xe0e0 : \
                        (0x0000) )

#define SHUFW_ELEM(x_) ( ((x_)>='A' && (x_)<='D') ? (0x00010203 + (0x04040404 * ((u32)(x_)-'A'))) : \
                        ((x_)>='a' && (x_)<='d') ? (0x10111213 + (0x04040404 * ((u32)(x_)-'a'))) : \
                        ((x_)=='0') ? (0x80808080) : \
                        ((x_)=='x' || (x_)=='X') ? (0xc0c0c0c0) : \
                        ((x_)=='8') ? (0xe0e0e0e0) : \
                        (0x00000000) )
```

```
#define SHUFB(b0_,b1_,b2_,b3_,b4_,b5_,b6_,b7_,b8_,b9_,b10_,b11_,b12_,b13_,b14_,b15_) \
(qword) (vu8){ SHUFB_ELEM(b0_), SHUFB_ELEM(b1_), SHUFB_ELEM(b2_), SHUFB_ELEM(b3_), \
SHUFB_ELEM(b4_), SHUFB_ELEM(b5_), SHUFB_ELEM(b6_), SHUFB_ELEM(b7_), \
SHUFB_ELEM(b8_), SHUFB_ELEM(b9_), SHUFB_ELEM(b10_), SHUFB_ELEM(b11_), \
SHUFB_ELEM(b12_), SHUFB_ELEM(b13_), SHUFB_ELEM(b14_), SHUFB_ELEM(b15_), \
}
```

```
#define SHUFH(h0_,h1_,h2_,h3_,h4_,h5_,h6_,h7_) \
(qword) (vu16){ SHUFH_ELEM(h0_), SHUFH_ELEM(h1_), SHUFH_ELEM(h2_), SHUFH_ELEM(h3_), \
SHUFH_ELEM(h4_), SHUFH_ELEM(h5_), SHUFH_ELEM(h6_), SHUFH_ELEM(h7_), \
}
```

```
#define SHUFW(w0_,w1_,w2_,w3_) \
(qword) (vu32){ SHUFW_ELEM(w0_), SHUFW_ELEM(w1_), SHUFW_ELEM(w2_), SHUFW_ELEM(w3_), }
```

```
#define SHUF(s_) ((sizeof(#s_)==5) ? SHUFW(#s_[0], #s_[1], #s_[2], #s_[3]) : \
(sizeof(#s_)==9) ? SHUFH(#s_[0], #s_[1], #s_[2], #s_[3], \
#s_[4], #s_[5], #s_[6], #s_[7]) : \
(sizeof(#s_)==17) ? SHUFB(#s_[0], #s_[1], #s_[2], #s_[3], \
#s_[4], #s_[5], #s_[6], #s_[7], \
#s_[8], #s_[9], #s_[10], #s_[11], \
#s_[12], #s_[13], #s_[14], #s_[15]) : \
SHUFW('0', '0', '0', '0'))
```

```
#define DEF_SHUF(s_) qword shuf_##s_ = ((sizeof(#s_)==5) ? SHUFW(#s_[0], #s_[1], #s_[2], #s_[3]) : \
(sizeof(#s_)==9) ? SHUFH(#s_[0], #s_[1], #s_[2], #s_[3], \
#s_[4], #s_[5], #s_[6], #s_[7]) : \
(sizeof(#s_)==17) ? SHUFB(#s_[0], #s_[1], #s_[2], #s_[3], \
#s_[4], #s_[5], #s_[6], #s_[7], \
#s_[8], #s_[9], #s_[10], #s_[11], \
#s_[12], #s_[13], #s_[14], #s_[15]) : \
SHUFW('0', '0', '0', '0'))
```

////////////////////////////////////

helper macros contd.

- so the actual usefulness of these is debatable as we get a lot of crud in `static_initialization_and_destruction`
- but they're quite cool nonetheless!

end!

- questions ?

