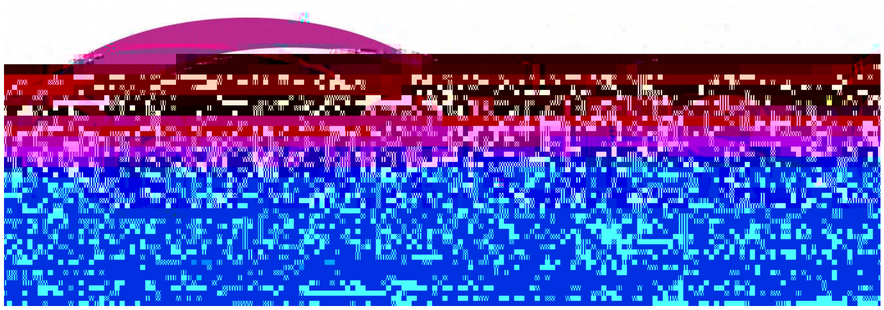

Editor: Robert J. Simpson, Qualcomm

OpenGL GLSL editor: John Kessenich, LunarG
GLSL version 1.1 Authors: John Kessenich, Dave Baldwin, Randi Rost

!

*.(.1 !............. nBaPaloI

-his document speci'ies onl. version (.1 o' the OpenGL ES Shadin& Lan&ua&e. $t re=uires BB1ERS$O : BB to substitute (1+, and re=uires to accept onl. (1+ es. $' is declared #ith a smaller number, the lan&ua&e accepted is a previous version o' the shadin& lan&ua&e, #hich #ill be supported dependin& on t

- mod' 'unction
- Se=uence and ternar. operators #ith t.pe
- Se=uence and ternar. operators #ith arra. t.pes

- rra.s o' arra.s
- tomic counters
- $ma&es
- Separate pro&ram obGects @also 7no#n as separate shader obGectsA
-

$

,

-he source character set used 'or the OpenGL ES shadin& lan&ua&es is 9nicode in the 9-4;5 encodin& scheme. $nvalid 9-4;5 characters are i&nored. !urin& pre;processin&, the 'ollo#in& applies:

- b.te #ith the value Eero is al#a.s interpreted as the end o' the strin&
- "ac7slash @FJFA, is used to indicate line continuation #hen immediatel. precedin& a ne#;line.
-

-he source 'or a sin&le shader is an arra. o' strin&s o' characters 'rom the character set. sin&le shader is made 'rom the concatenation o' these strin&s. Each strin& can contain multiple lines, separated b. ne#; lines. : o ne#;lines need be present in a strin&L a sin&le line can be 'ormed 'rom multiple strin&s. : o

00 12'00 #ill substitute a decimal inte&er constant that is one more than the number o' precedin& ne#; lines in the current source strin&.

0041 '00 #ill substitute a decimal inte&er constant that sa.s #hich source strin& number is currentl. bein& processed.

00 ' (1$200 #ill substitute a decimal inte&er re'lectin& the version number o' the OpenGL ES shadin& lan&ua&e. -he version o' the shadin& lan&ua&e described in this document #ill have *00 ' (1$200* substitute the decimal inte&er (1+.

& 0'(#ill be de'ined and set to 1. -his is not true 'or the non;ES OpenGL Shadin& Lan&ua&e, so it can0 ` nI euqiAproper

*			. &
1 @hi&hestA	parenthetical &roupin&	@ A	:
)	unar.	de'ined N ; O P	Ri&ht to Le't
(	multiplicative	Q ? R	

allo#s implementation dependent compiler control. -o7ens 'ollo#in& are not sub6ect to preprocessor macro e2pansion. $' an implementation does not reco&niEe the to7ens 'ollo#in& , then it #ill i&nore that pra&ma. -he 'ollo#in& pra&mas are de'ined as part o' the lan&ua&e.

pra7id2.'otpi pra&-h0l2.'mAms ` 0S 1 ! sOm R otptlae2.rt oote lan&uam : n&

! !

0 1

-he lan&ua&e is a se=uence o' to7ens. to7en can be

,

, 3)

)

! @`

H H H
<? <?> B <? ! <? ! > B
<? <? ! <? <? !
E? E? > B E? E?
H H H
E? >! E? >! E? >!

E ; F
E E E ; E F
; E ; ; ; F
F E F ; F F
G ! G ! > B G !
G !
E? E? E?
G ! G ! G !
E? > E? > E? >
E? >! E? >! E? >!

) : one o'
$ < E ; F X Y Z [%

$denti'iers startin& #ith H&lBI are reserved 'or use b. OpenGL ES, and ma. not be declared in a shader. $t is an error to redeclare a variable, includin& those startin& H&lBI.

-he ma2imum len&th o' an identi'ier is 1+)* characters. $t is an error i' the len&th e2ceeds this value.

5 -

Some lan&ua&e rules described belo# depend on the 'ollo#in& de'initions.

5ò 5 › 0

shader contains a o' a variable . i', a'ter preprocessin&, the shader contains a statement that #ould read or #rite . @or part o' 2A, #hether or not run;time 'lo# o' contro Rnrtir

5 -& & 6) 5

! !

1(.

! (&

ll variables and 'unctions must be declared be'ore bein& used. 1ariable and 'unction names are

$ %& ! '(!

& 9

E?

-o ma7e conditional e2ecution o' code easier to e2press, the t.pe is supported. -here is no

E2amples

>	*?2)	4 >2
>	*?2 @	4 >2
5>	*?2 @ ; (	2 4 5>2
5>	*?2 @ ;	4 A
A BAAAAAAA	*?2 =:5	2
A B7 %&AA	*? =:5	2
A	*?2)	4 5>2
A CAAAAAAA	*?2 %	5:>9D9C=E9C2
A	*?2 @	4 A 2

! ' (

-he OpenGL ES Shadin& Lan&ua&e includes data t.pes 'or &eneric);, (;, and *;component vectors o' 'loatin&;point values, inte&ers, and "ooleans. 4loatin&;point vector variables can be used to store colors, normals, positions, te2ture coordinates, te2ture loo7up results and the li7e. "oolean vectors can be used 'or component;#ise comparisons o' numeric vectors. Some e2amples o' vector declarations are:

```
:        >4        :
=                  :
```

"ecause a sin&le opa=ue t.pe declaration e''ectivel. declares t#o obGects, the opa=ue handle itsel' and the obGect it is a handle to, there is room 'or both a stora&e =uali'ier and a memor. =uali'ier. -he stora&e =uali'ier #ill =uali'. the opa=ue handle, #hile the memor. =uali'ier #ill =uali'. the obGect it is a handle to.

! 2

Sampler t.pes @e.&., E?A are opa=ue t.pes, declared and behavin& as described above 'or opa=ue t.pes.

! 4

n arra . t.pe can be 'ormed b . speci' .in& a non;arra . t.pe @

$ %& ! '(!

M M 1 A 3 >A LL

0

6

)
0

6

0
)) 1) A) ;
))
6

1 % ; 2

! : 7

E2amples o' combinations that are disallo#ed:

1.

```
02226
02226
```

$ %& ! '(!

! ;

: ote that 'unction parameters can use , , and =uali'iers, but as % ! .
3arameter =uali'iers are discussed in section 0.1.1 @H4unction %allin& %onventionsIA. 4unction return

!)

.% is one o'

- a literal value @e.&. X or

- / ! %
- n % %
- n arra.
- structure

E2ample declarations in a verte2 shader:

9
=

-he output o' the verte2 shader and the input o' the 'ra&ment shader 'orm a shader inter'ace. 4or this compile or lin7 error is &enerated i' an. o' the e2plicitl. &iven or compiler &enerated uni'orm locations is &reater than the implementation;de'ined ma2imum number o' uni'orm locations minus one.

9nli7e locations 'or inputs and outputs, uni'orm locations are lo&ical values, not re&ister locations, and

n inter'ace bloc7 is started b. a or 7e.#ord, 'ollo#ed b. a bloc7 name, 'ollo#ed b. an open curl. brace @0A as 'ollo#s:

/! ,

$' an instance name @ A

4or bloc7s declared as arra.s, the arra. inde2 must also be included #hen accessin& members, as in this e2ample

```
      ,       0    BK#      ,       I:J                  :
     9         H    '   H
     9         H    '   K  8     H
               -
 6        I9J
 222
```

! !) !
! !)
! !) ! !) !

! !)
! !
! ! > ! ! !

r. ‰`• 72ài-`hè¢o7ems"ám an. ! !) ! are identi'iers, not 7e.#ord0Eàot °

A T T N H D R H D ! B N

;E
<Y
;E
[
[C

;E
ima&e t.pes
<Y
onl.
[
;E

;E
<Y
[
;E

! ! & ;

ll shaders e2cept compute shaders allo# input la.out =uali'iers on input variable declarations. -he la.out =uali'ier identi'ier 'or shader inputs is:

! !)
6

Onl. one ar&ument is accepted. 4or e2ample,

```
;          1 =     9
```

#ill establish that the shader input *!*

; ;! !

Speci'.in& this #ill ma7e per;'ra&ment tests be per'ormed be'ore 'ra&ment shader e2ecution. $n addition it is an error to staticall. #rite to &lB4ra& ! epth in the 'ra&ment shader. $' this is not declared, per;

-he location speci'ies the location b. #hich the OpenGL ES 3$ can re'erence the uni'orm and update its

; 4 ! 8
; 4 ! 8

$' the /)

-he identi'ier bindin& speci'ies #hich unit #ill be bound. n. uni'orm sampler, ima&e or atomic counter variable declared #ithout a bindin& =uali'ier is initiall. bound to unit Eero. 'ter a pro&ram is lin7ed, the unit re'erenced b. a sampler uni'orm variable declared #ith or #ithout a bindin& =uali'ier can be updated b. the OpenGL ES 3$.

" indin& points are not inherited, onl. o''sets. Each bindin&

```
  !                 !
          ;E
          <Y
       ;E
          [
          [C
                  !
          ;E
          <Y
          [
       ;E
                   !
          ;E
          <Y
          [
       ;E
```

'ormat la.out =uali'ier speci'ies the ima&e 'ormat associated #ith a declared ima&e variable. Onl. one 'ormat =uali'ier ma. be speci'ied 'or an. ima&e variable declaration. 4or ima&e variables #ith 'loatin&; point component t.pes @

-he re=uired ran&es and precisions 'or precision =uali'iers are:

; + + * + *
* : 9 * :
:
6

•

! 4)

$nvariance must be &uaranteed 'or constant e2pressions. particular constant e2pression (possible [illegible]

'   )

-he constructor . / preserves the bit pattern in the ar&ument, #hich #ill chan&e the ar&ument's value i' its si&n bit is set. -he constructor . / preserves the bit pattern in the ar&ument, #hich #ill chan&e its value i' it is ne&ative.

Some use'ul vector constructors are as 'ollo#s:

```
=                         .                                    =
9      9          <          9                  5
9     :                   9              A              ;          >

:        4                                        .          :        :
 =     4     4                                    .            =        =
 9     4     4       4                      9 7

:     =                                                            =
=     9                                                             9

=     :4                 =2   1     :2 4     =2; 1     :2;4     =2. 1
=       4     :          =2   1       4     =2; 1     :2 4     =2. 1     :2;
9     =4
9       4     =
9     :4     :
```

Some e2amples o' these are:

```
9        1     9 A2A4 >2A4 A2A4 >2A
9        1     9 >2A                                   >2A
=        1     =                      9
```

-o initialiEe the dia&onal o' a matri2 #ith all other elements set to Eero:

```
:
=
9
```

-hat is, *! 7 87‹8* is set to the 'loat ar&ument 'or all > ‹

-o initialiEe a matri2 b. speci'.in& vectors or scalars, the components are assi&ned to the matri2 elements in column;maCor order.

```
:      :4     :
=      =4     =4     =
9      94     94     94     9
= :       :4     :4     :

:          4          4
           4

=          4          4          4
           4          4          4
           4          4                 !)à! à

9          4          4          4          4
           4          4          4          4
           4          4          4          4
           4          4          4

: =       :4          4
          :4
```

#ide ran&e o' other possibilities e2ist, to construct a matri2 'rom vectors and scalars, as lon& as enou&h

' ! ! . &

rra. t.pes can also be used as constructor names, #hich can then be used in e2pressions or initialiEers. 4or e2ample,

!) !! !

!) !! !

#here the &eneral e2pression

5 1

!   ) !! !

°

!) !! !

•

'   "   (9   )

nd

```
  =  4  4

 1   M
```

is e=uivalent to

```
IAJ2  1  IAJ2  M  IAJ2   L   I>J2  M  IAJ2;  L   I:J2  M  IAJ2.
I>J2  1  IAJ2  M  I>J2   L   I>J2  M  I>J2;  L   I:J2  M  I>J2.
I:J2  1  IAJ2  M  I:J2   L   I>J2  M  I:J2;  L   I:J2  M  I:J2.

IAJ2; 1  IAJ2; M  IAJ2   L   I>J2; M  IAJ2;  L   I:J2; M  IAJ2.
I>J2; 1  IAJ2; M  I>J2   L   I>J2; M  I>J2;  L   I:J2; M  I>J2.
I:J2; 1  IAJ2; M  I:J2   L   I>J2; M  I:J2;  L   I:J2; M  I:J2.

IAJ2. 1  IAJ2. M  IAJ2   L   I>J2. M  IAJ2;  L   I:J2. M  IAJ2.
I>J2. 1  IAJ2. M  I>J2   L   I>J2. M  I>J2;  L   I:J2. M  I>J2.
I:J2. 1  IAJ2. M  I:J2   L   I>J2. M  I:J2;  L   I:J2. M  I:J2.
```

and similarl. 'or other siEes o' vectors and matrices.

')

-he %NN standard re=uires that e2pressions must be evaluated in the order speci'ied b. the precedence o' operations and ma. onuns an% ! p

0

%!
) !
.%
!

‹ %

Simple declaration, e2pression, and Gump statements end in a semi;colon.

-his above is sli&htl. simpli'ied, and the complete &rammar speci'ied in section 6 HShadin& Lan&ua&e GrammarI should be used as the de'initive speci'ication.

!eclarations and e2pressions have alread. been discussed.

0 + -

*   !

*   !

- he t.pe o'

1 A 3 >A LL
0

6

-he **B** loop 'irst e2ecutes the bod., then e2ecutes the *)* .% . -his is re,eteesR €-s'hds àA36atad m,Ji)

*   !

2 , 8 (

2 , 8 (

2

2

4ra&ment shader helper invocations e2ecute the same shader code as non;helper invocations, but #ill not

-he built;in constant !0 J ,& %(5 is a compute;shader constant containin& the local #or7;&roup

2 , 8

-he 'ollo#in& built;in const st

2   ,   8   6

& + , !

4 . & +

4unction parameters speci'ied as ! are assumed to be in units o' radians. $n no case #ill an. o' these

&)	-
&en4 - . pe @&en4 - . pe . A	Returns the h . perbolic sine 'unction
	. _ ‾.

&) **-**

hi&hp &en4-.pe @hi&hp &en4-.pe ., in hi&hp &en$-.pe .%AL	-he 'unction lde2p@A builds a sin≤precision 'loatin&; point number 'rom each si&ni'icand component in . and dds('á Q"ic& int&bal e2pone2nt Rp< o in

.%

4 ! + 8* * 1 6 1 +

&)

&)

& +   ,   !

&   )   -

&en4 - . pe

40 9)+

&) - g)

4 2 (: +

Relational and e=ualit. op=u 1 0t. op=u 1 0tb !

4 4 +

&) **-**

&en$ - . pe **K** @&en$ - . pe *!* , E2tracts bits thrŏu&h
int , int / AL

&en 9 - . pe **K** @&en 9 - . pe *!* ,
int , int / AL

&) -

lo#p &en$-.pe **A>H**@&en$-.pe ! AL Returns the bit number o' the least si&ni'icant one bit in
lo#p &en$-.pe **A>H**@&en9 -.pe ! AL

4 5) +

-e2ture loo7up 'unctions are available in all shadin& sta&es. , o#ever, level o' detail is implicitl. computed onl. 'or 'ra&ment shaders. Other shaders operate as thou&h the base level o' detail #ere

4 5) 1 +

&)

&) -

[redacted]

&vec* ple (! S IR @&sampler) ! %! , vec(-,
(ivec)], 'loat / ^ A
&vec* pl (! S IR @&(sampler) ! %! , vec* -,
ivec)], 'loat / ^ A
- %! &Qe* #0 c(S IR @&sampler(! %! , vec* -,
ivec( ], 'loat / ^ A
@saR "3 lĐHnQRV 1 3 c(pp
SR 'loat S IR @sampler) ! Shado# %! * vec$] @ %WRp ((

[redacted]

Wló òĐ

4or te2ture &ather 'unctions usin& a shado# sampler t.pe, each o' the 'our te2el loo7ups per'orm a depth comparison a&ainst the depth re'erence value passed in @re'YA, and returns the result o' that comparison in the appropriate component o' the result vector.

s #ith other te2ture loo7up 'unctions, the results o' a te2ture &ather are unde'ined 'or shado# samplers i' the te2ture re'erenced is not a depth te2ture or has depth comparisons disabledL or 'or non;shado# samplers i' the te2ture re'erenced is a depth te2ture #ith depth comparisons enabled.

-he @ **R** built;in 'unctions 'rom the OpenGL ES Shadin& Lan&ua&e return a vector derived 'rom samplin& 'our te2els in the ima&e arra. o' level *! !0/* . 4or each o' the 'our te2el o''sets speci'ied b. the

&) -

&vec*

& + , !

4 " . 8 +

4   .   9   & +

& + , !

&) -

uint

&)		-
hi&hp ivec)	> @readonl. #riteonl. &ima&e) ! ima&eA	Returns the dimensions o' the ima&e or ima&es bound to . 4or arra.ed ima&es, the last component o' the return value #ill hold the siEe o' the arra.. %ube ima&es onl. return the dimensions o' one 'ace.
hi&hp ivec(	> @readonl. #riteonl. &ima&e(! ima&eA	
hi&hp ivec)	> @readonl. #riteonl. &ima&e%ube ima&eA	
hi&hp ivec(	> @readonl. #riteonl. &ima&e) ! rra. ima&eA	: ote: -he =uali'ication **B** accepts a variable =uali'ied #ith , **B** , both, or neither. $t means the 'ormal ar&ument #ill be used 'or neither readin& nor #ritin& to the underl.in& memor..
hi&hp &vec*	**A** @readonl.	

"ac7#ard di''erencin&:

$4 \;. -). \; - 4 \;. \;\; -)4). \;. \;). \qquad)a$

$)4). \;. \quad \dfrac{4 \;. \; - 4 \;. -).}{).} \qquad)b$

> ith sin≤sample rasteriEation,). SU 1.+ in e=uations 1b and)b. 4orU)b. 7uçlpeĕ riEation

& +   ,   !

4or compute shaders, the @A 'unction ma. be placed #ithin control 'lo#, but that control 'lo# must

& + , !

5 **9**

s described in chap

" -

5

```
"%&,!KB(%$ (#+Q,!KB(%$ "%&,!7(BN?%, (#+Q,!7(BN?%, "%&,!7(BN% (#+Q,!7(BN% -*,
N*HHB N*"*$ %W@B" )%H#N*"*$ 7B$+ -B)Q ,#"-% K"@) ),B( )"B)Q K%(N%$,
"%&,!B$+"% (#+Q,!B$+"% '%(,#NB"!7B( NB(%, BHK%()B$- W@%),#*$
```

-he 'ollo#in& describes the &rammar 'or the OpenGL ES Shadin& Lan&ua&e in terms o' the above to7ens.

/! 0)
1 '2"141'

% 0 .%
/! 0)
12"+$2("A2"
N12"+$2("A2"
4 $A"+$2("A2"
B$$ +$2("A2"
'4"0-A '2 .% 1&H"0-A '2

% .0 .%
% 0 .%
% .0 .% '4"0B A+P'" 0 .% 1&H"0B A+P'"
0 !!
% .0 .% $" 41' 0(' '+"1$2
% .0 .% 12+0$-
% .0 .% '+0$-
?? 41' 0(' '+"1$2

* *

0 !!0#) 03 #0%

0 !!0#) 0 .%

0 !!0#) 03 #0% +$ = =A 0 .%

0 !!0#)

0) '4"0-A '2

II & 2 + ! , ! , * / ! . !

)) 0 .% A(H ! %! 0 .%

 # 0 .%

)) 0 .%

 # 0 .% '4"0$ -)) 0 .%

 # 0 .% 1&H"0$ -)) 0 .%

! !0 .%

 # 0 .%

! !0 .% '4"0A2& ' # 0 .%

! !0 .% 1&H"0A2& ' # 0 .%

! !0. 0 .%
! !0 0 .% $ 0$ - ! !0. 0 .%

) !0 .%
! !0 0 .%
! !0 0 .% TN ' ("1$ 2 .% + $ $ 2 0 .%

0 .%
) !0 .%
0 .% 0 % 0 .% $ 2

0 %
' TNA
= N 0A((1&2
1 0A((1&2
= $ 0A((1&2
A 0A((1&2
(NB0A((1&2
3 ' 4"0A((1&2) N '4 0p $1 yóW$ $ 2
1&H"0A((1&2
A2 0A((1&2
K$ 0A((1&2
$ 0A((1&2

.%
3) 0N '4 0p $1 yóW$ $ 2
3 .%) N$ 4 = A0p 1$2 yóW$ $ 2 + $ $ 2 @Nóp "0 .% 0% 1 1 !0 .%

* *

% 0 ! (' =1+$ $2

\# * *

À ¾

N(A= - '
N(A= - '
N(A= - ' +NB '
N(A= - ' A AL
S < 3LER) ! < S
$S < 3LER) ! < S
9S < 3LER) ! < S
S < 3LER) ! < S RR D
$S < 3LER) ! < S RR D
9S < 3LER) ! < S RR D
$< GE) !
$$< GE) !
9$< GE) !
$< GE(!
$$< GE(!
9$< GE(!
$< GE%9 "E
$$< GE%9 "E
9$< GE%9 "E
$< GE) ! RR D
$$< GE) ! RR D
9$< GE) ! RR D
0 %
"L - ' 02A= '

% 0 !
H1&H0 - ' +1(1$ 2
= ' 1N= 0 - ' +1(1$ 2
$ J 0 - ' +1(1$ 2

0 %
"1'4"° " 0) 0(" 1&H0+ '2"14'1' '4"° " 0)

0) ! 0!
0) !
0) ! 0! +$==A 0) !

0) !
1 '2"141'
1 '2"141' 0 %

!5
0 .%

) ! 0
) !

%)0 03 #0 %
%! 0 %

0 0 30 %
%)0 0AW1 '2$A yF=óp %! 0

%)0 0 0 30 %
' 4 "0B A+ ' 1&H"0B A+ '
' 4 "0B A+ ' 0! 1&H"0B A+ '

0!

0!

.% 0
(' =1+ $ $ 2
.% (' =1+ $ $ 2

! 0
14 ' 4 "0-A ' 2 .% 1&H"0-A ' 2 ! 0 0

! 0 0
P0)0 0 0 Ᵽ 0 ⁻ό€ Đ

* *

S+++(: %onditional Gump parameter @ , , **B** , ;**B** A must be a boolean

!

!

L+++(: -oo man. verte2 input values

- : on;s=uare matrices o' t.pe mat%2R consume the same space as a s=uare matri2 o' t.pe mat : #here : is the &reater o' % and R. 1ariables o' t.pe mat) occupies) complete ro#s. -hese

.	!	!

E2ample: pac7 the 'ollo#in& t.pes:

9

Option: : o, this #ould be e2pensive to implement on devices that do not nativel. support it.

RESOL9 -$O : : : o. -he speci'ication should allo# e''icient implementation o' mediump 'loat on 10;bit 'loatin& point hard#are but must also be implementable on devices #hich onl. nativel. support ();bit 'loatin& point.

Should the 'ra&ment shader have a de'ault precisionC

1erte2 shaders have a de'ault hi&h precision because lo#er precisions are not su''icient 'or the maGorit. o' &raphics applications. , o#ever, man. 'ra&ment shader operations do not bene'it 'rom hi&h precision and

, o# should the roundin& mode be speci'iedC

< ost current implementations support round;to;nearest. Some but not all also support round;to;nearest; even.

RESOL9 -$O : : > ithin the accurac. speci'ication, the roundin& mode should be unde'ined.

Should there be &EPp rnEaenp(i rvWestirtsEme a00c &irmEmhcacderPeKp@ion,,

Option): : o.

!!   !

2 + - + %

Should local 'unctions hide all 'unctions o' the same sameC

-his is considered use'ul i' local 'unction declarations are allo#ed. , o#ever, the onl. use 'or local 'unction declarations in GLSL ctic0sips Q1Q l,ctions hide oons cR tG tr •` lr G°'ltb lQasEptinsESll ±63pôàÙlrations are

0 6

0
1 >2A +")" %)4 < +")"2
6

RESOL9 -$O : : Keep the basic preprocessor as de'ined in the GLSL ES 1.++ speci'ication.

4

Option: Replace the current version directive mechanism #ith a b.te or character se=uence that must al#a.s occur at the start o' the shader. -his is similar to other standards that have multiple versions e.&. , --3.

Option: <a7e 9-4;5 characters an optional 'eature o' GLSL ES 1.++

!! !

!! !

Option 1: Speci' . all operations to be invariant. : o, this is too restrictive. Optimum use o' resources

! !   !

$mplementations must usuall. be able to evaluate constant e2pressions at compile time since the. can be

!! !

!! !

E2ample:

0

6

222

1

Option 1: n unde'ined value is returned to the caller. :o error is &enerated. -his is #hat most cNN compilers do in practice @althou&h the cNN standard actuall. speci'ies Funde'ined behaviorFA.

Option): -here must be a return statement at the end o' all 'unction de'initions that return a value.

:o, this re=uires statements to be added that ma. be impossible to e2ecute.

Option B: return statement at the end o' a 'unction de'inition is re=uired onl. i' it is possible 'or e2ecution to reaches the end o' the 'unction:

E.&.

0

>

A

$ 2 ,

2

6

'iip@hePssi#le to e2e `n s m1 @ be

0

> hat compiler trans'orms should be allo#edC

5 9 : /

GLSL ES 1.++ speci'ied a set o' minimum re=uirements that e''ectivel. made parts o' the speci'ication

!   :   1

Should an error be &enerated i' a literal inte&er is outside the ran&e o' a ( ) ;bit inte&erC

One possibilit. is to chan&e the de'inition o' the se=uence operator so that it does not return a constant; e2pression and hence cannot be used to declare an arra. siEe.

```
          IGJ
              1A     3 2             LL
      I J 1 A2A
```

Option): Si&ned inte&er. -his allo#s &reater 'le2ibilit. in calculatin& arra. indices #ithout the need 'or t.pe conversions e.&.

```
          I)#Z%J
   222
             1   2          5 =      "      ;      2 )#Z%    ;          <

             X1  A                            <
        I      J
```

RESOL9 -$O : : Option). -he principle is that inte&ers that represent values and hence ma. 'orm part o' arithmetic e2pressions should al#a.s be si&ned, even i' it is 7no#n that the. #ill al#a.s be positive. 1alues that represent bit vectors should al#a.s be unsi&ned.

-he e2tra chec7in& made available b. the use o' unsi&ned inte&ers 'or values 7no#n to be positive is

-here are some valid uses 'or aliasin&.   n Fuber shaderF @i.e. a lar&e shader that consists o' multiple selectable smaller shadersA mi&ht have too man. verte2 inputs i' the. all have uni=ue locations but could

!4

```
M  I=J 1          I:JI=J
```

Option (:

Replace the current post'i2 arra. t.pes #ith a pre'i2 notation:

```
I:JI=J             B   ;       .  :          ;       .  =
```

-he t.pe speci'ier is similar to the s.nta2 in other lan&ua&es #here declarations ta7e the 'orm:

is arra. *5* o' */* *%*

Option 1: Des and each 'unction has one si&nature. -here is no attempt to overload each 'unction based

! . 1 $

- his speci'ication is based on the #or7 o' those #ho contributed to the OpenGL ES (.+ Lan&ua&e Speci'ication, the OpenGL ES).+ Lan&ua&e Speci'ication, and the 'ollo#in& contributors to this version:

John Kessenich, LunarG

Jon Kenned., (! Labs

Jon Leech, Khronos

Jonathan 3utsman, $ma&ination -echnolo&ies

Joohoon Lee, Samsun&

Jou7oK.lmco6a, S.mbio

Jrn :.stad, R<

Jussi Rasanen, :1$!$

Kalle Raita, dra#Elements

Kari 3ulli, :o7ia

' 7 à

- he OpenGL d ER, Graphics R, .stem 1ersion (.1+
- he OpenGL d ER, R, pp, hRpcppe rsion (.1 " p,++ ! àÑ