```
(define sqrt-iter
  (delayed-propagator
   (lambda (x g answer)
     (let ((done (make-cell))
           (not-done (make-cell))
           (x-again (make-cell))
           (g-again (make-cell))
           (new-g (make-cell))
           (new-answer (make-cell)))
       (good-enuf? g x done)
       (switch done g answer)
       (inverter done not-done)
       (switch not-done new-answer answer)
       (switch not-done x x-again)
       (switch not-done g g-again)
       (heron-step x-again g-again new-g)
       (sqrt-iter x-again new-g new-answer)))))
```